Getting Started

- (1) Copy to your hard disk from a USB Key or DVD:
  - Copy needed files (VirtualBox, terminal, possibly an X server) for your platform (Win/Mac/Linux)
  - Copy Java 6 and Eclipse for your platform, if you want to use Java
  - **Copy VM image: OpenFlowTutorial-101311.zip**
  - Pass on the DVD or USB key to someone else!

- (2) Unzip OpenFlowTutorial-101311.zip

- (3) Point browser to instructions:
  - http://www.openflow.org/wk/index.php/OpenFlow_Tutorial (note the underscore)

- You should NOT need to download any large files – spare the WiFi!

OpenFlow Hands-on Tutorial

part of the the Open Networking Summit
Li Ka Shing Center
Stanford University
Oct 17, 2011

Brandon Heller
Stanford University

with help from all the people
listed on the next few pages

This tutorial wouldn't be possible without:

- OpenFlow Experts
  - Glen Gibb
  - Nicholas Bastin
  - Ali Al-Shabibi
  - Tatsuya Yabe
  - Masayoshi Kobayashi
  - Yiannis Yiakoumis
  - Ali Yahya
  - Te-Yuan Huang
  - Bob Lantz
  - David Erickson

This tutorial wouldn't be possible without:

- Deployment Forum Speakers
  - Subhasree Mandal (Google)
  - Johan van Reijendam (Stanford)
  - David Erickson (Stanford)
- Videographer:
  - Yiannis Yiakoumis

This tutorial wouldn't be possible without:

- Past slides from:
  - Nick McKeown
  - Rob Sherwood
  - Guru Parulkar
  - Srini Seetharaman
  - Yiannis Yiakoumis
  - Guido Appenzeller
  - Masa Kobayashi, + others

Welcome

Getting Started

- (1) Copy to your hard disk from a USB Key or DVD:
  - Copy needed files (VirtualBox, terminal, possibly an X server) for your platform (Win/Mac/Linux)
  - Copy Java 6 and Eclipse for your platform, if you want to use Java
  - **Copy VM image: OpenFlowTutorial-101311.zip**
  - Pass on the DVD or USB key to someone else!

- (2) Unzip OpenFlowTutorial-101311.zip

- (3) Point browser to instructions:
  - http://www.openflow.org/wk/index.php/OpenFlow_Tutorial (note the underscore)

- You should NOT need to download any large files – spare the WiFi!

Agenda

Time	Description
9:30-10:45	Introduction: Motivation, History, Interface
11:00-12:00	What can you do with OpenFlow and SDN?
1:30-2:30	SDN Building Blocks
2:45-4:00	SDN Deployment Panel and Wrap-up

feel free to:

- ask any kind of OpenFlow question during the hands-on
- take breaks at any point during the hands-on
- work during the 15-minute breaks, or lunch

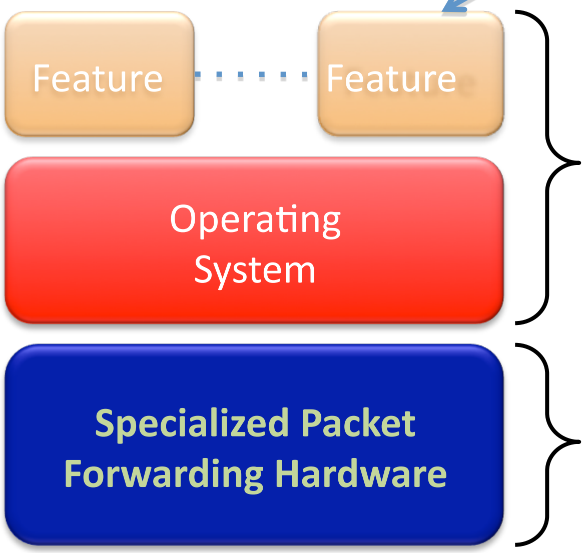
Goals of this Tutorial

- By the end, everyone should know:
  - **what** OpenFlow is
  - **how** it's used and how *you* can use it
  - **where** it's going
  - how OpenFlow relates to Software-Defined Networking (SDN)
- Present a useful mix of hands-on and lecture-based content
- **Have fun**

Why OpenFlow?

The Ossified Network

Routing, management, mobility management, access control, VPNs, …



Million of lines of source code   5400 RFCs   Barrier to entry

Billions of gates   Bloated   Power Hungry

Many complex functions baked into the infrastructure

OSPF, BGP, multicast, differentiated services,
Traffic Engineering, NAT, firewalls, MPLS, redundant layers, …

An industry with a “mainframe-mentality”, reluctant to change

Research Stagnation

- Lots of *deployed* innovation in other areas
  - OS: filesystems, schedulers, virtualization
  - DS: DHTs, CDNs, MapReduce
  - Compilers: JITs, vectorization
- Networks are largely the same as years ago
  - Ethernet, IP, WiFi
- Rate of change of the network seems slower in comparison
  - Need better tools and abstractions to demonstrate and deploy

Closed Systems (Vendor Hardware)

- Stuck with interfaces (CLI, SNMP, etc)
- Hard to meaningfully collaborate
- Vendors starting to open up, but not usefully

Open Systems

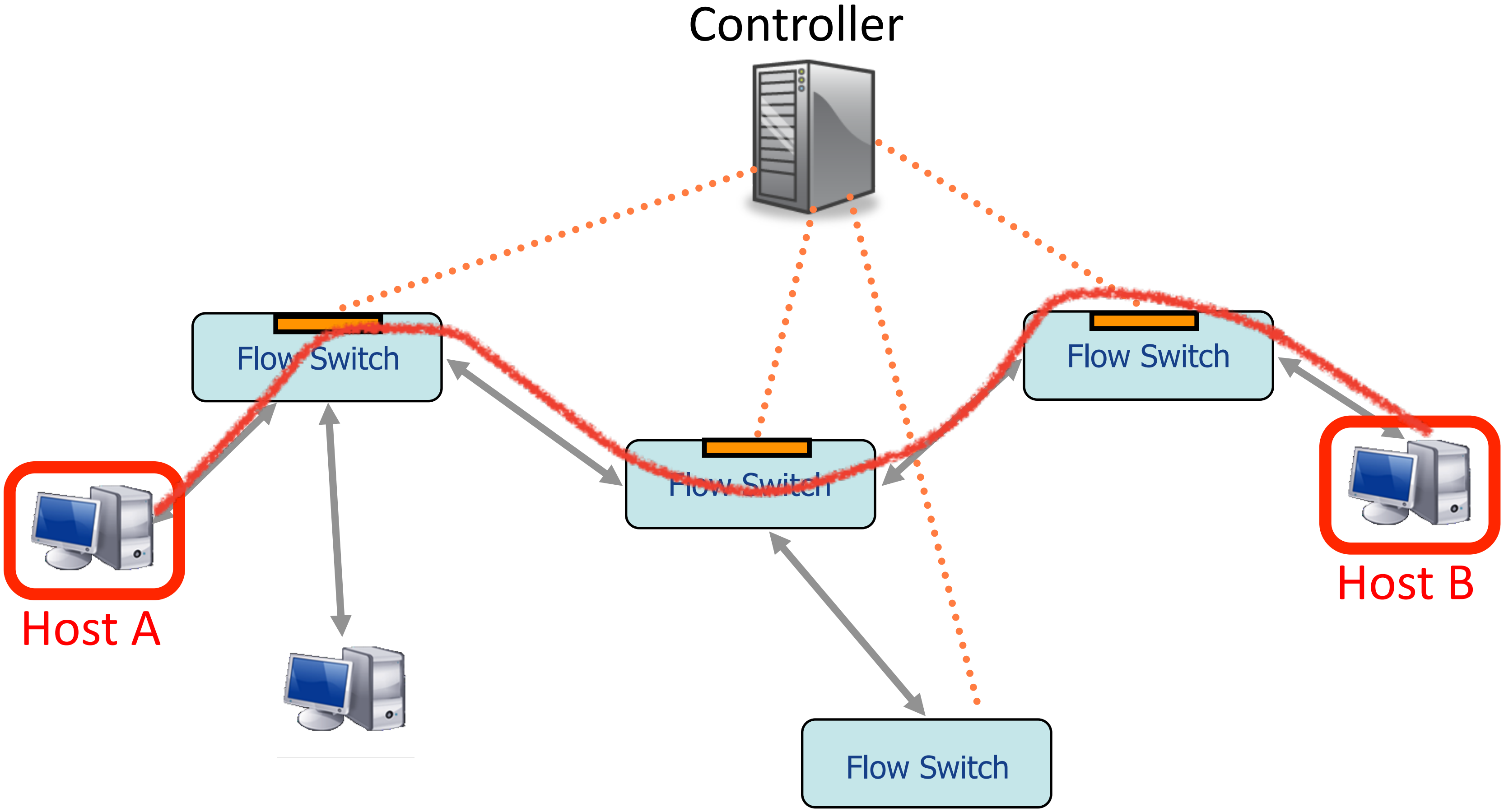
	Performance Fidelity	Scale	Real User Traffic?	Complexity	Open
Simulation	medium	medium	**no**	medium	yes
Emulation	medium	**low**	**no**	medium	yes
Software Switches	**poor**	**low**	yes	medium	yes
NetFPGA	high	**low**	yes	**high**	yes
Network Processors	high	medium	yes	**high**	yes
Vendor Switches	**high**	**high**	**yes**	**low**	**no**

gap in the tool space

none have **all** the desired attributes!

Ethane, a precursor to OpenFlow

Centralized, reactive, per-flow control



See Ethane SIGCOMM 2007 paper for details

OpenFlow: a pragmatic compromise

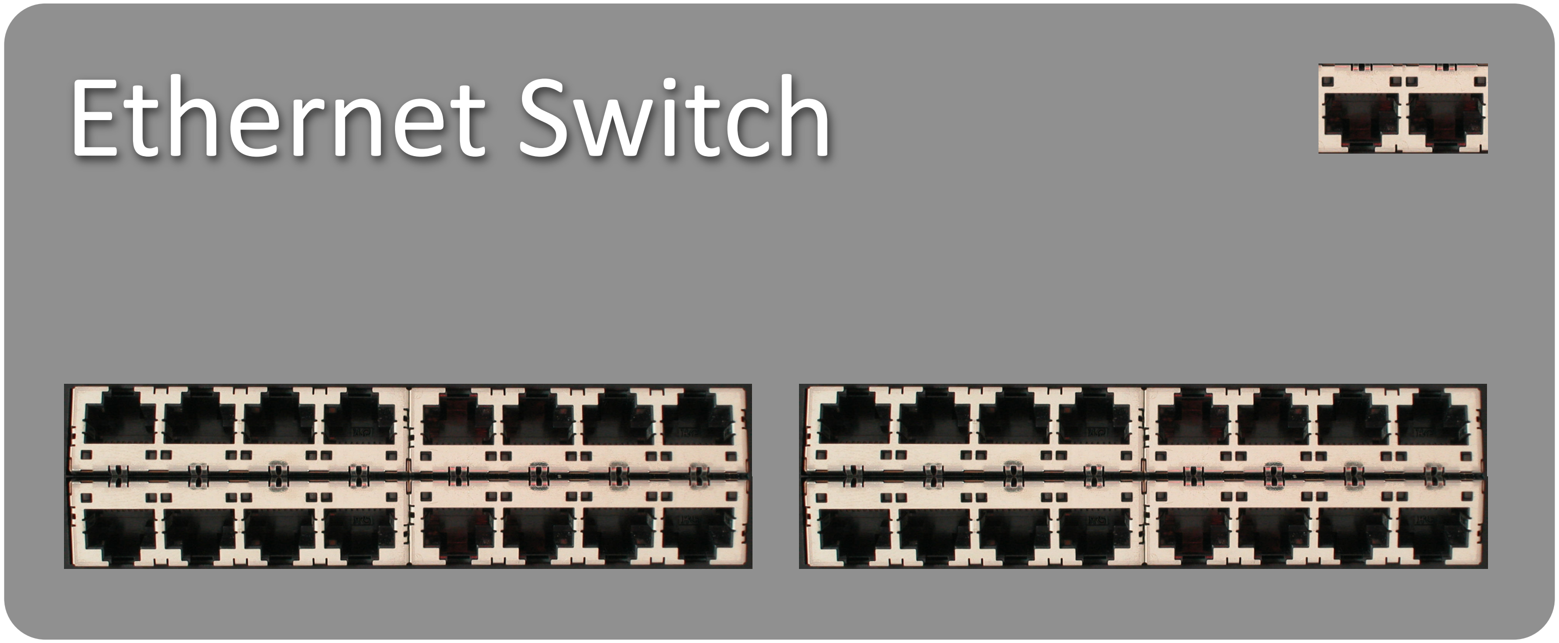
- + Speed, scale, fidelity of vendor hardware
- + Flexibility and control of software and simulation
- Vendors don't need to expose implementation
- Leverages hardware inside most switches today (ACL tables)

It’s hard to add a feature to a network

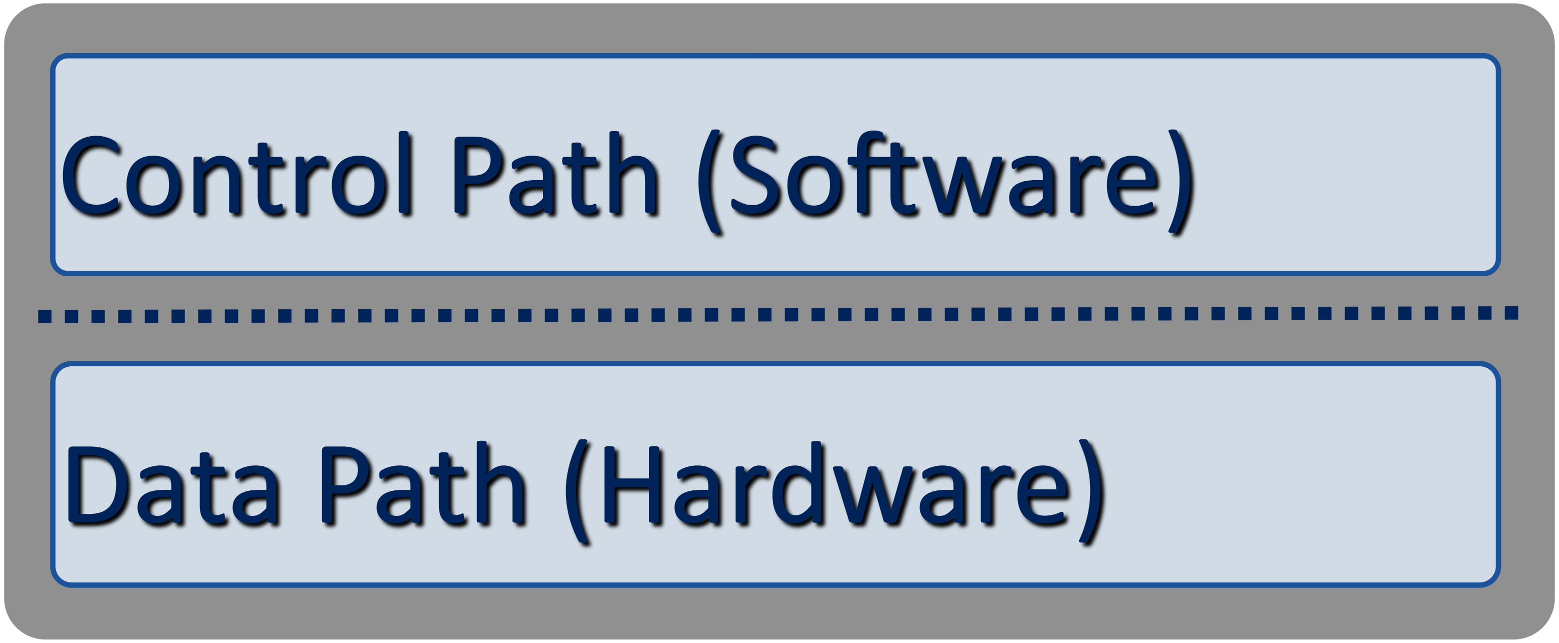
- It’s not just that we lack access to line-rate forwarding that we can control
- Fully distributed algorithms are hard, especially when defined at the protocol level
- Your protocol must implement its own mechanisms
- Must work on constrained and heterogeneous resources

This is where Software-Defined Networking comes in. More to come later today.

How does OpenFlow work?

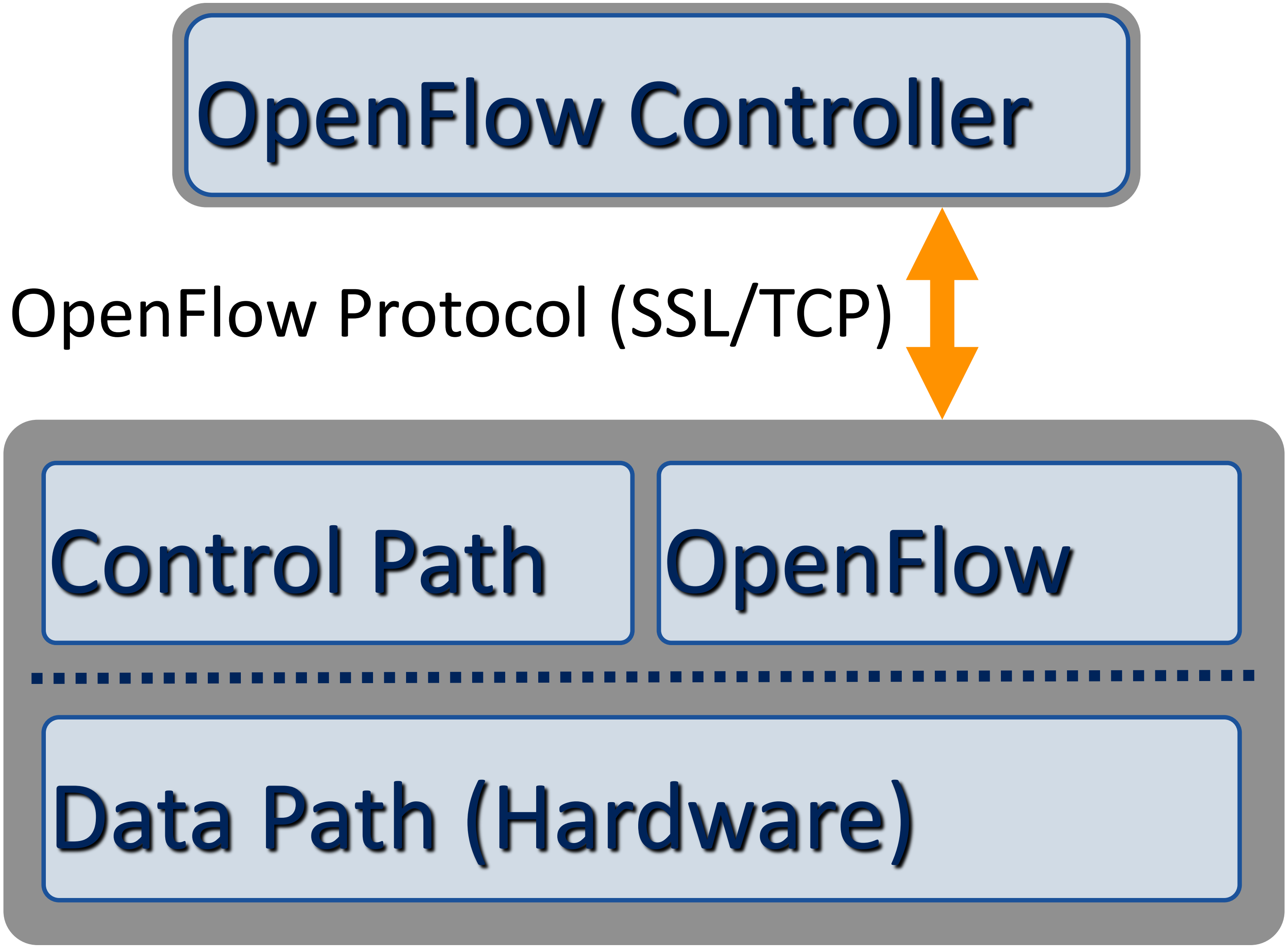


Ethernet Switch



Control Path (Software)

Data Path (Hardware)



OpenFlow Controller

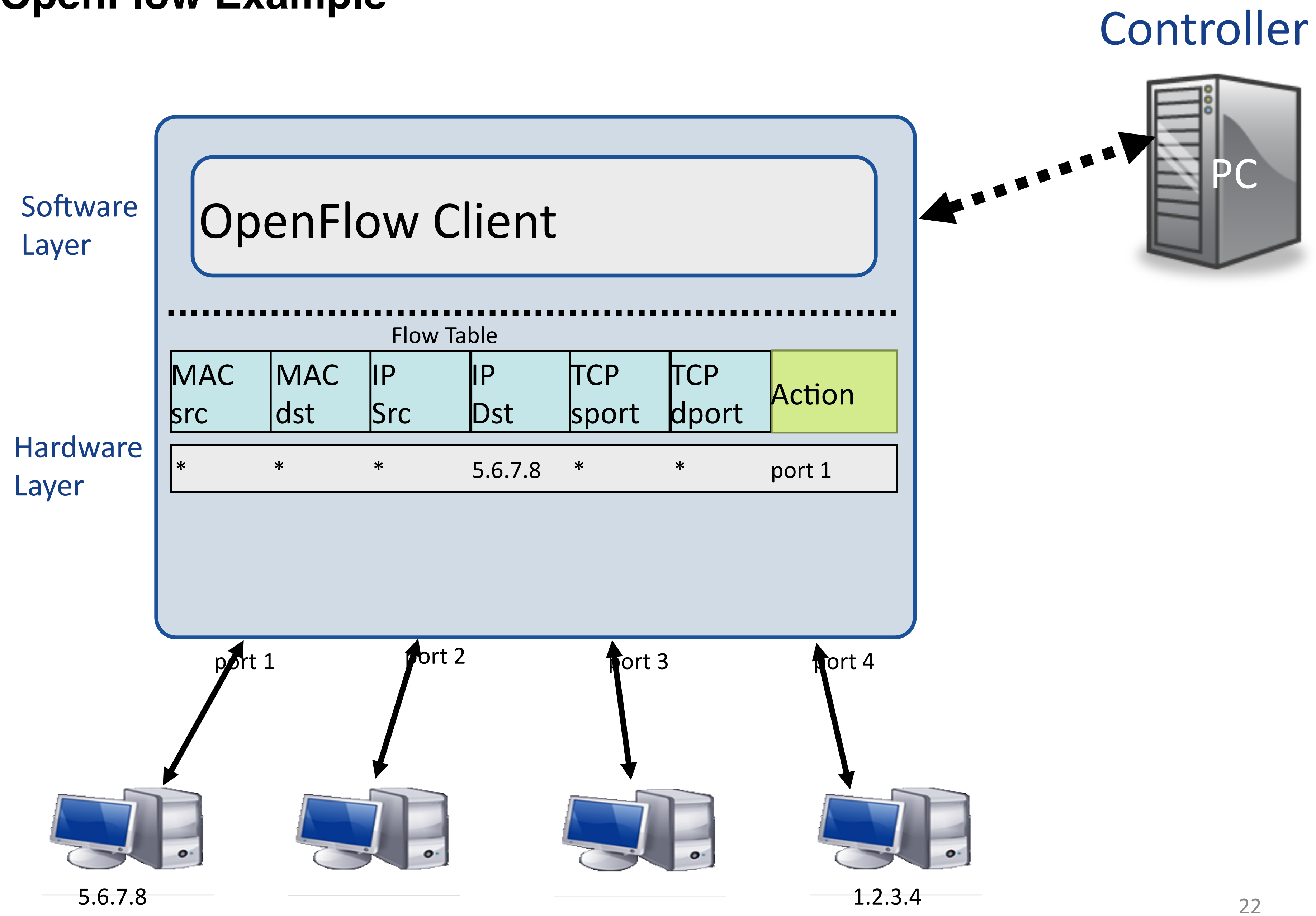
OpenFlow Protocol (SSL/TCP)

Control Path

OpenFlow

Data Path (Hardware)

OpenFlow Example



Controller

PC

Software Layer

OpenFlow Client

Hardware Layer

Flow Table

MAC src	MAC dst	IP Src	IP Dst	TCP sport	TCP dport	Action
*	*	*	5.6.7.8	*	*	port 1

port 1

port 2

port 3

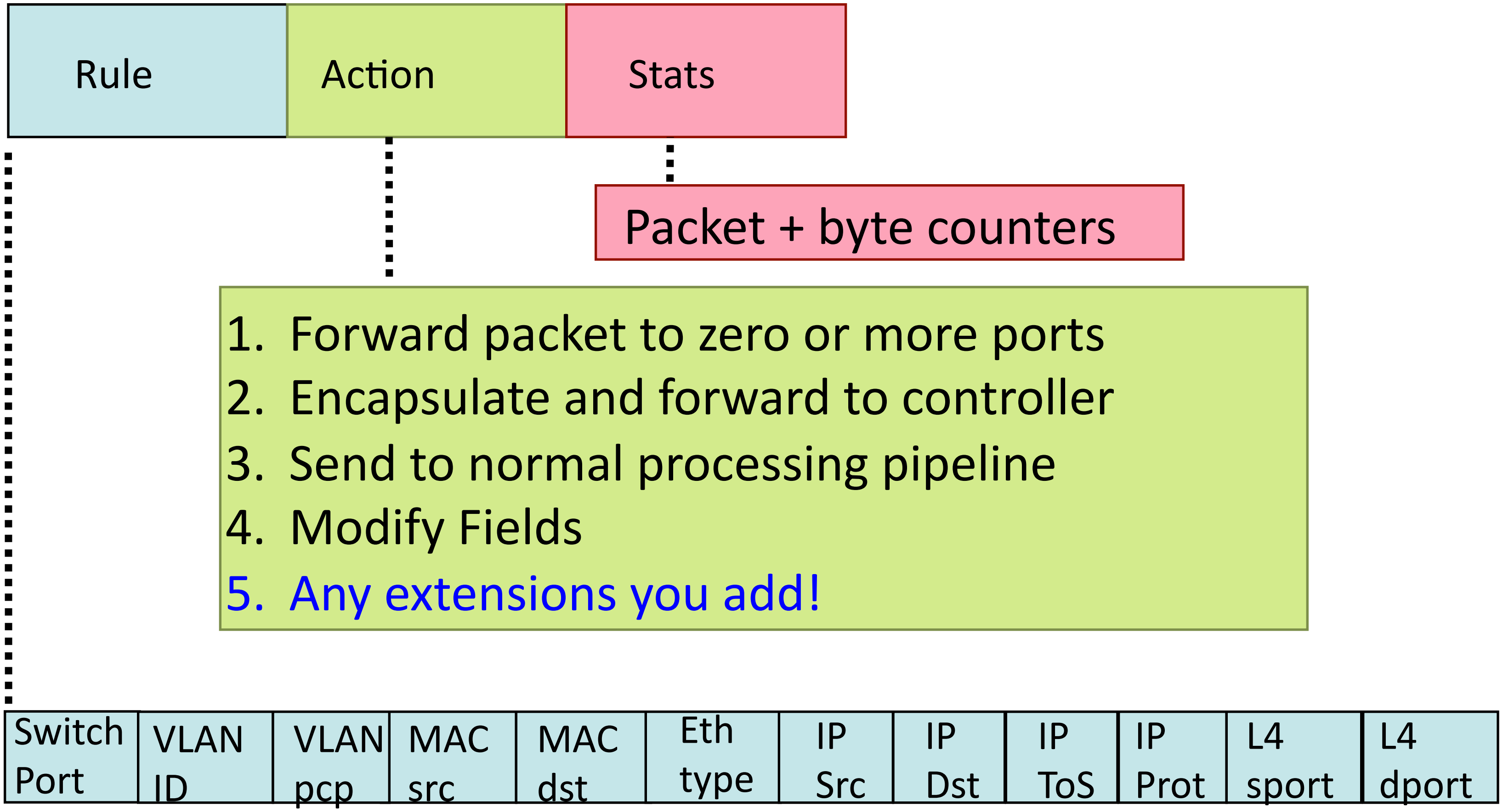
port 4

5.6.7.8

1.2.3.4

OpenFlow Basics

Flow Table Entries



Rule	Action	Stats

Packet + byte counters

1. Forward packet to zero or more ports
2. Encapsulate and forward to controller
3. Send to normal processing pipeline
4. Modify Fields
5. Any extensions you add!

Switch Port	VLAN ID	VLAN pcp	MAC src	MAC dst	Eth type	IP Src	IP Dst	IP ToS	IP Prot	L4 sport	L4 dport

+ mask what fields to match

Examples

Switching

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	00:1f:..	*	*	*	*	*	*	*	port6

Flow Switching

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
port3	00:20..	00:1f..	0800	vlan1	1.2.3.4	5.6.7.8	4	17264	80	port6

Firewall

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	*	*	*	*	*	*	*	22	drop

Examples

Routing

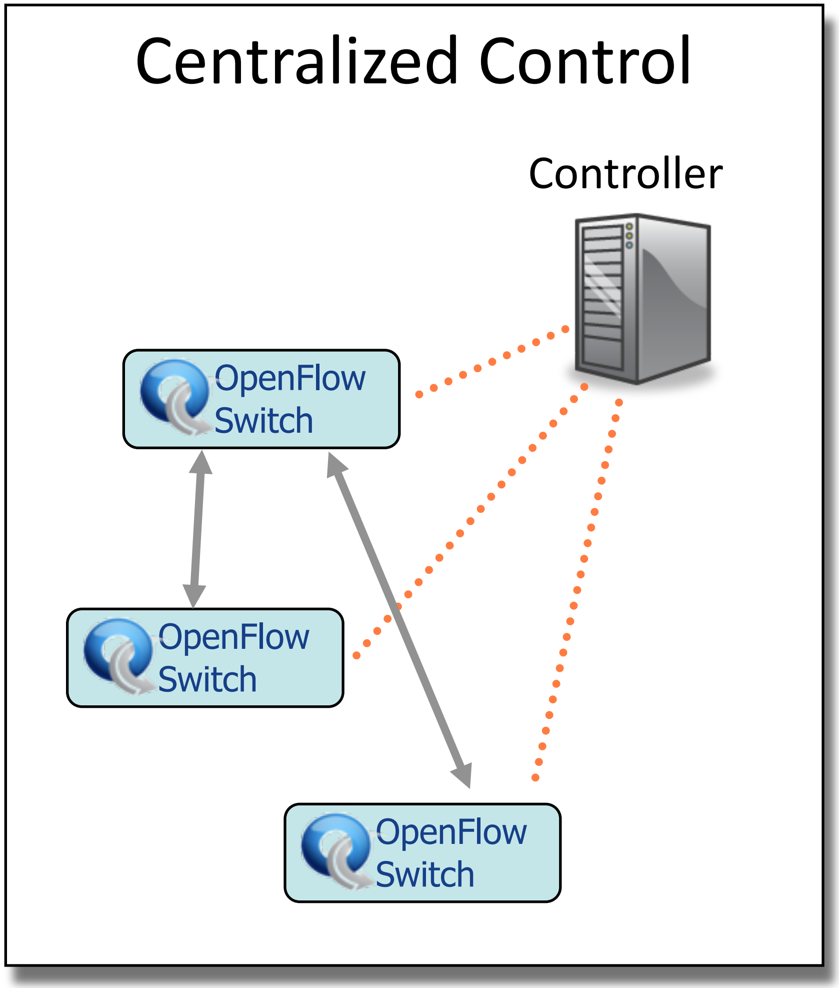
Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	*	*	*	*	5.6.7.8	*	*	*	port6

VLAN Switching

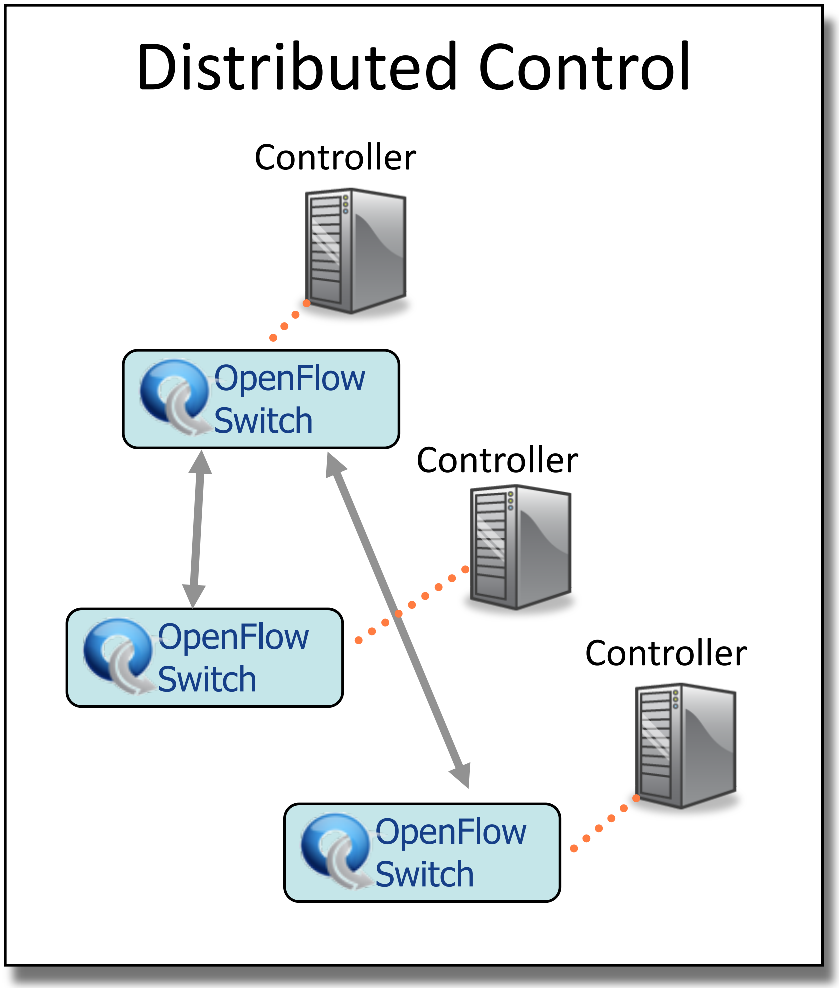
Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	00:1f..	*	vlan1	*	*	*	*	*	port6, port7, port9

Centralized vs Distributed Control

Anything in this spectrum is possible with OpenFlow



Centralized Control



Distributed Control

Flow Routing vs. Aggregation

Anything in this spectrum is possible with OpenFlow

Flow-Based

- Every flow is individually set up by controller
- Exact-match flow entries
- Flow table contains one entry per flow
- Good for fine grain control, e.g. campus networks

Aggregated

- One flow entry covers large groups of flows
- Wildcard flow entries
- Flow table contains one entry per category of flows
- Good for large number of flows, e.g. backbone

Reactive vs. Proactive (pre-populated)

Anything in this spectrum is possible with OpenFlow

Reactive

- First packet of flow triggers controller to insert flow entries
- Efficient use of flow table
- Every flow incurs small additional flow setup time
- If control connection lost, switch has limited utility

Proactive

- Controller pre-populates flow table in switch
- Zero additional flow setup time
- Loss of control connection does not disrupt traffic
- Essentially requires aggregated (wildcard) rules

What you cannot do with OpenFlow v1.0

- Non-flow-based (per-packet) networking
  - ex. Per-packet next-hop selection (in wireless mesh)
  - yes, this is a fundamental limitation
  - BUT OpenFlow can provide the plumbing to connect these systems
- Use all tables on switch chips
  - yes, a major limitation (cross-product issue)
  - BUT OF version 1.1 exposes these, providing a way around the cross-product state explosion

What can cannot do with OpenFlow v1.0

- New forwarding primitives
  - BUT provides a nice way to integrate them through extensions
- New packet formats/field definitions
  - BUT a generalized OpenFlow (2.0) is on the horizon
- Optical Circuits
  - BUT efforts underway to apply OpenFlow model to circuits
- Low-setup-time individual flows
  - BUT can push down flows proactively to avoid delays

Where it's going

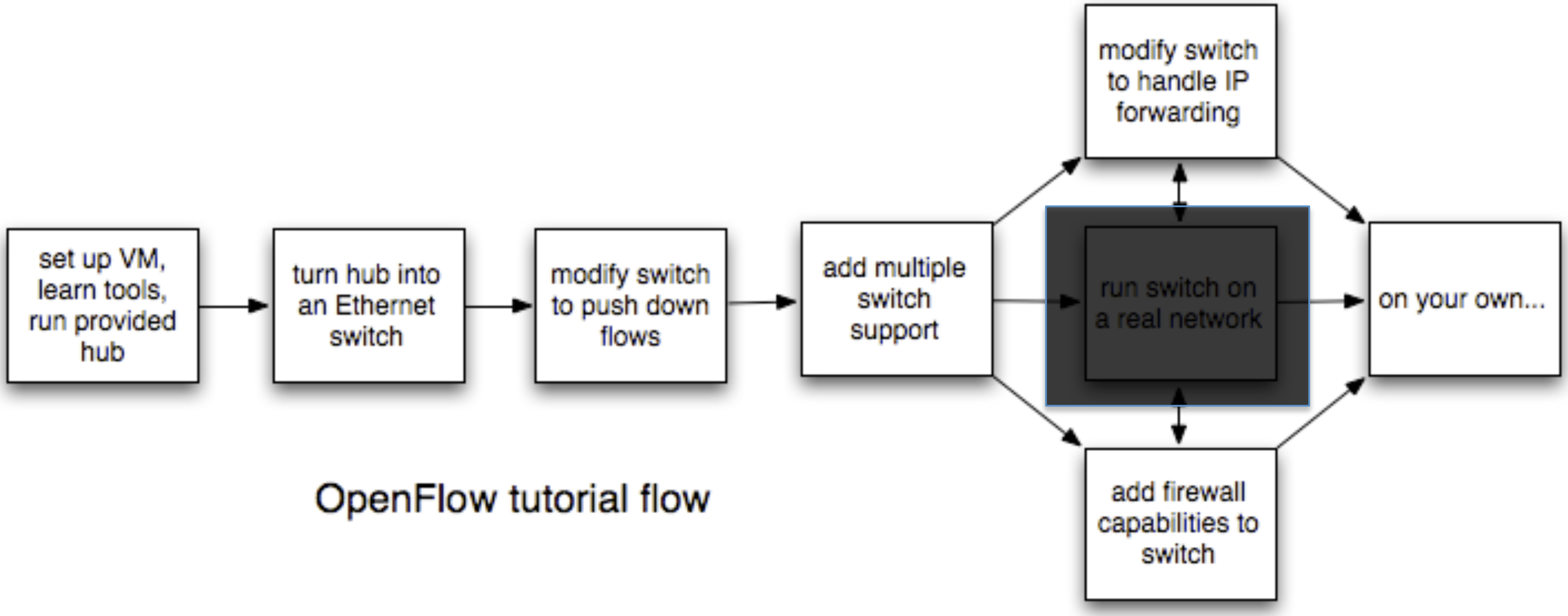
- OF v1.1: released March 1
  - multiple tables: leverage additional tables
  - tags and tunnels
  - multipath forwarding
- OF v1.2+
  - extensible match
  - generalized matching and actions: an "instruction set" for networking

Questions to think about

- How do I provide control connectivity?
- What are the flow setup times in OpenFlow networks?
- How can one take down an OpenFlow network?
- How can I perform topology discovery over OpenFlow-enabled switches?
- What happens when you have a non-OpenFlow switch in between?
- How scalable can an OpenFlow network be? How does one scale deployments?

[Hands-on Tutorial] Overview

TutorialFlow


set up VM, learn tools, run provided hub
turn hub into an Ethernet switch
modify switch to push down flows
add multiple switch support
modify switch to handle IP forwarding
run switch on a real network
add firewall capabilities to switch
on your own...

OpenFlow tutorial flow

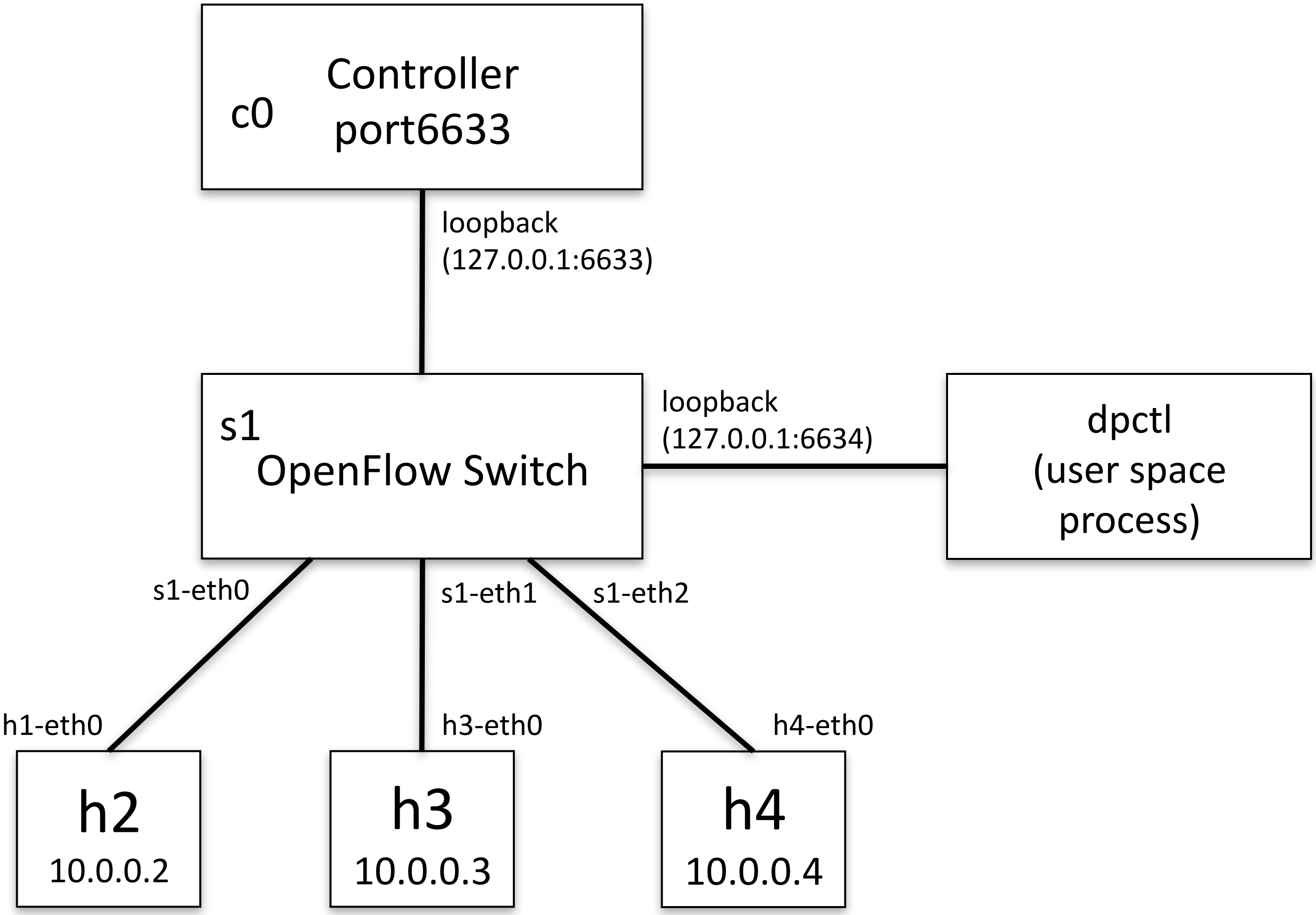
Stuff you'll use

- NOX
- Beacon
- Reference Controller/Switch
- Open vSwitch
- Mininet

- iperf
- tcpdump
- Wireshark

Tutorial Setup

OpenFlow Tutorial
3hosts-1switch
topology



virtual hosts

Hands-on Tutorial

Next presentation starts at **11:00:**

What can you do with OpenFlow / SDN?

Instructions still at:

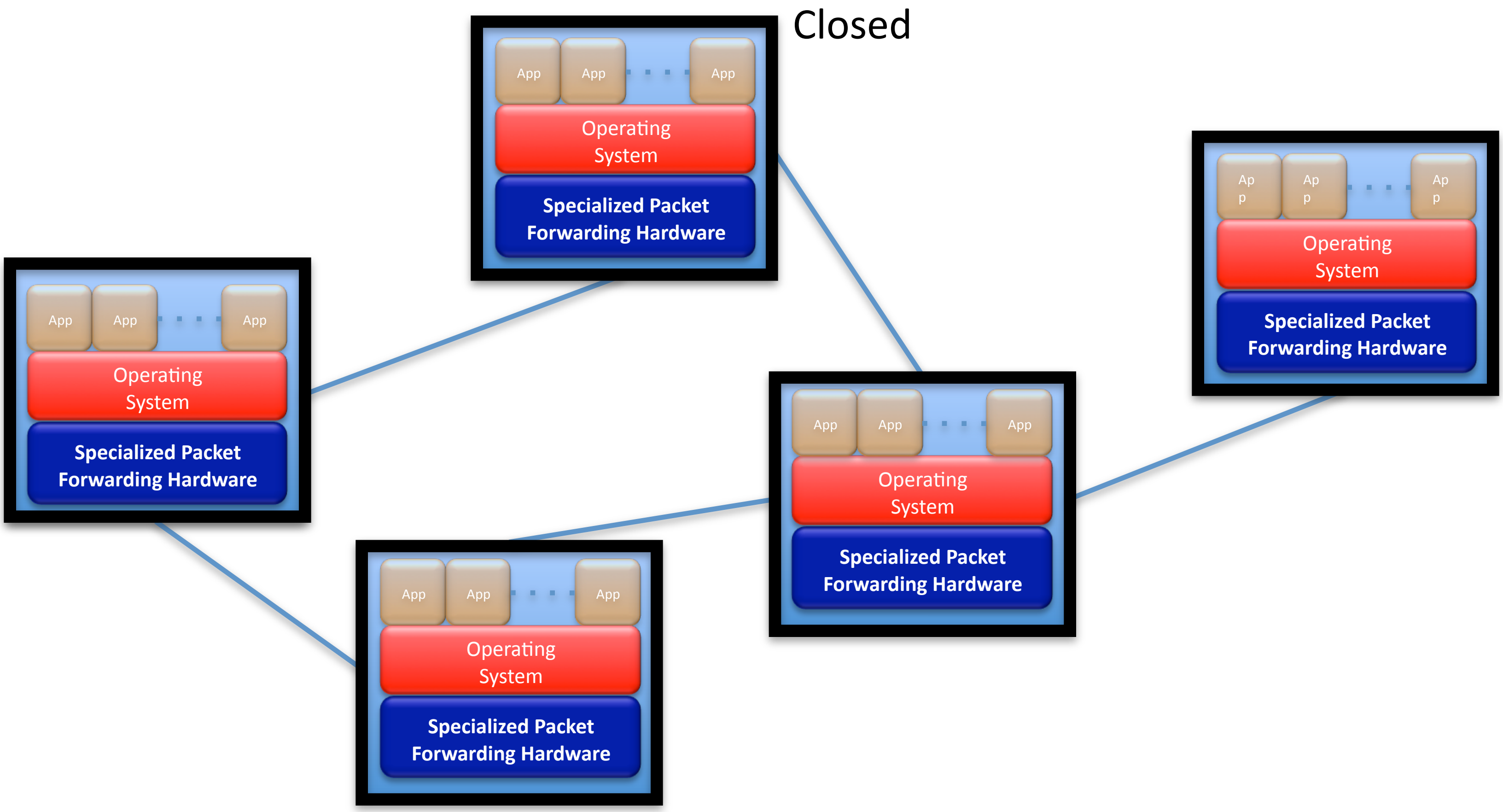
www.openflow.org/wk/index.php/OpenFlow_Tutorial

What can you do with OpenFlow/ SDN?

What is Software-Defined Networking (SDN)?

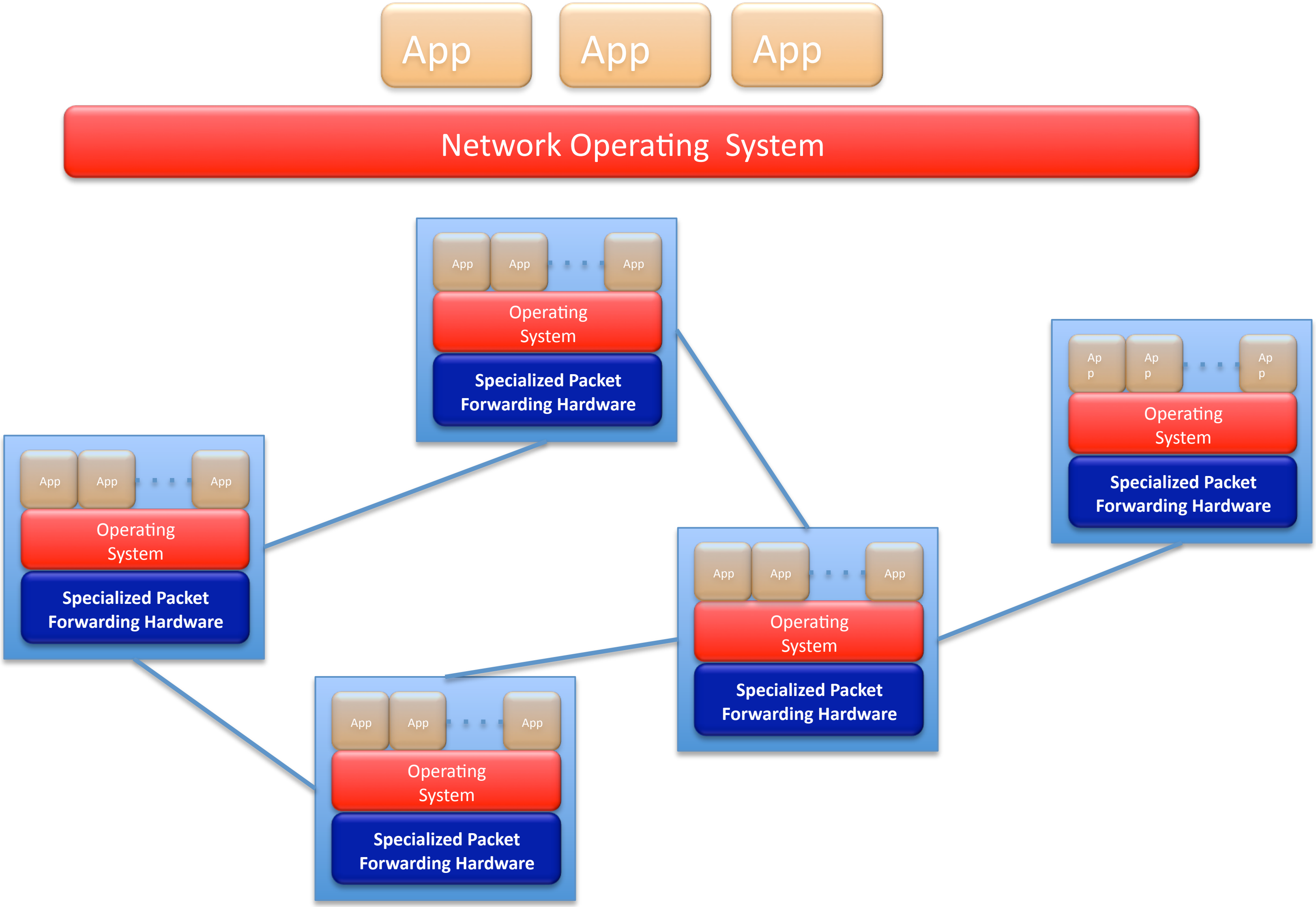
Status Quo

Closed Boxes, Fully Distributed Protocols

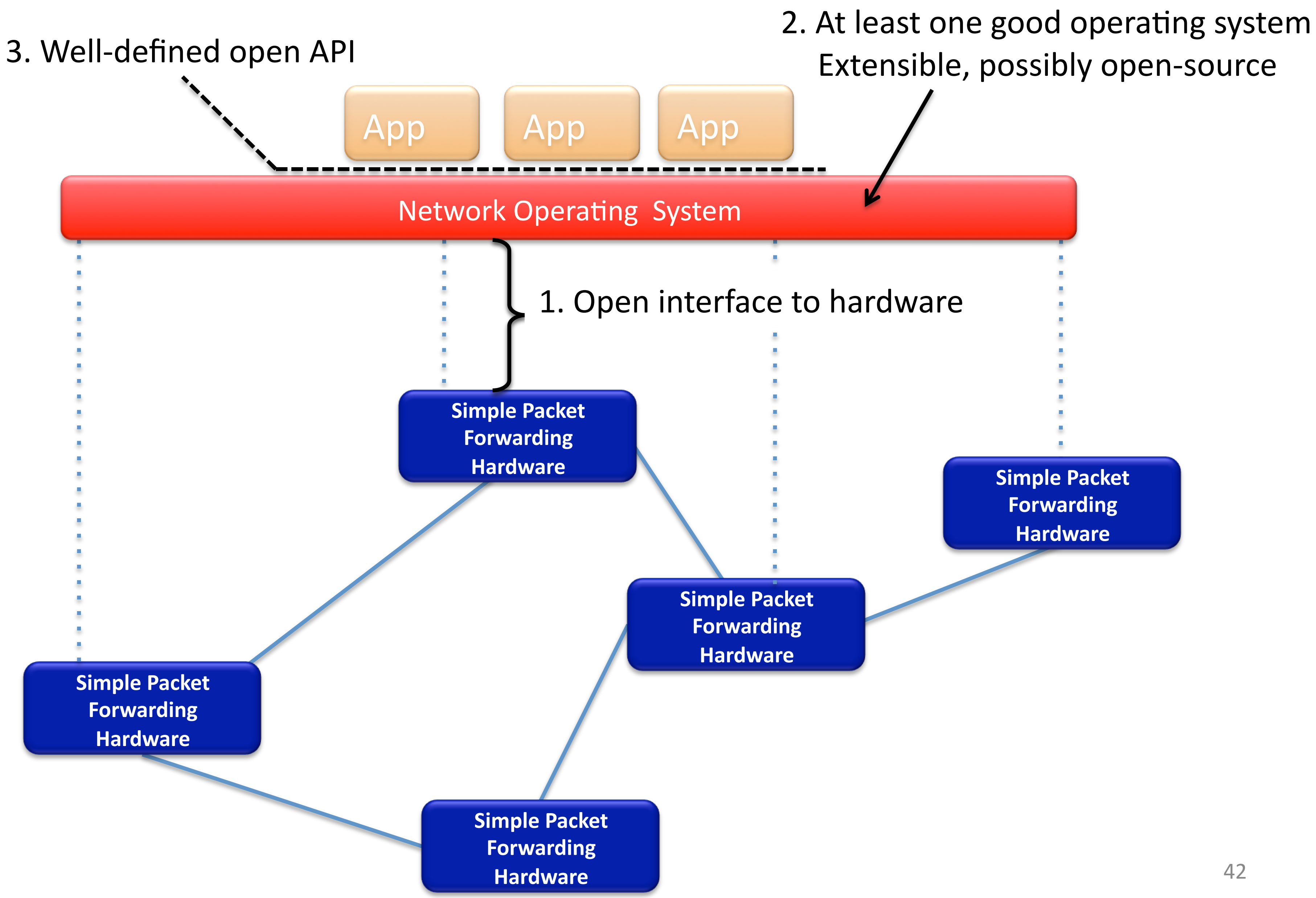


Closed
App
App
App
Operating System
Specialized Packet Forwarding Hardware
Ap p
Ap p
Ap p
Operating System
Specialized Packet Forwarding Hardware
App
App
App
Operating System
Specialized Packet Forwarding Hardware
App
App
App
Operating System
Specialized Packet Forwarding Hardware
App
App
App
Operating System
Specialized Packet Forwarding Hardware

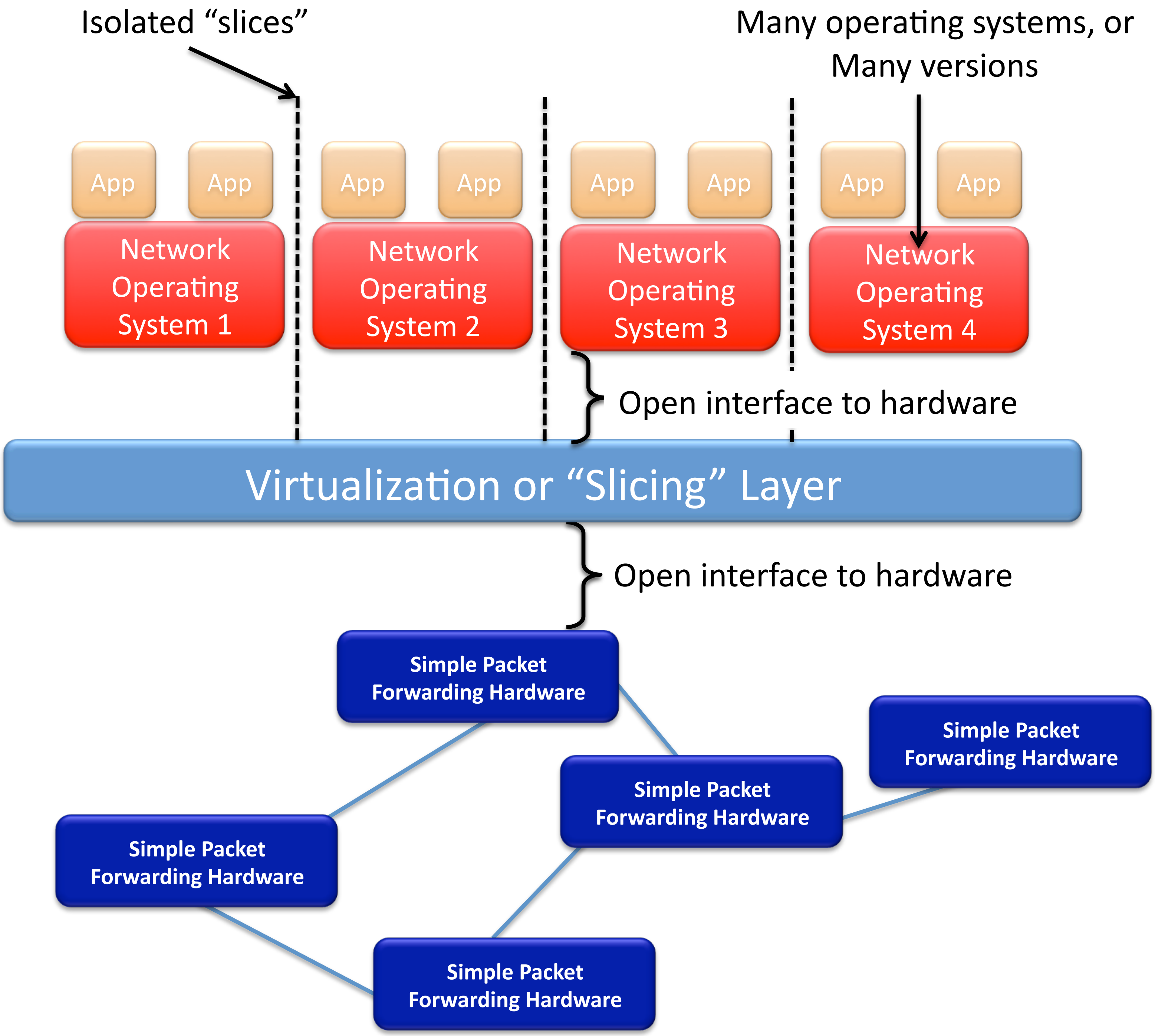
“Software Defined Networking” approach to open it



The “Software-defined Network”



3. Well-defined open API
2. At least one good operating system
Extensible, possibly open-source
App
App
App
Network Operating System
1. Open interface to hardware
Simple Packet Forwarding Hardware
Simple Packet Forwarding Hardware
Simple Packet Forwarding Hardware
Simple Packet Forwarding Hardware
Simple Packet Forwarding Hardware



Isolated “slices”

Many operating systems, or
Many versions

App App App App App App App App

Network Operating System 1

Network Operating System 2

Network Operating System 3

Network Operating System 4

Open interface to hardware

Virtualization or “Slicing” Layer

Open interface to hardware

Simple Packet Forwarding Hardware

Simple Packet Forwarding Hardware

Simple Packet Forwarding Hardware

Simple Packet Forwarding Hardware

Simple Packet Forwarding Hardware

SDN examples using OpenFlow

Simple primitive, but many applications

Stanford Demos

- Wireless mobility
- VM mobility/migration
- Network virtualization
- Power management
- Hardware
- Load balancing
- Traffic Engineering

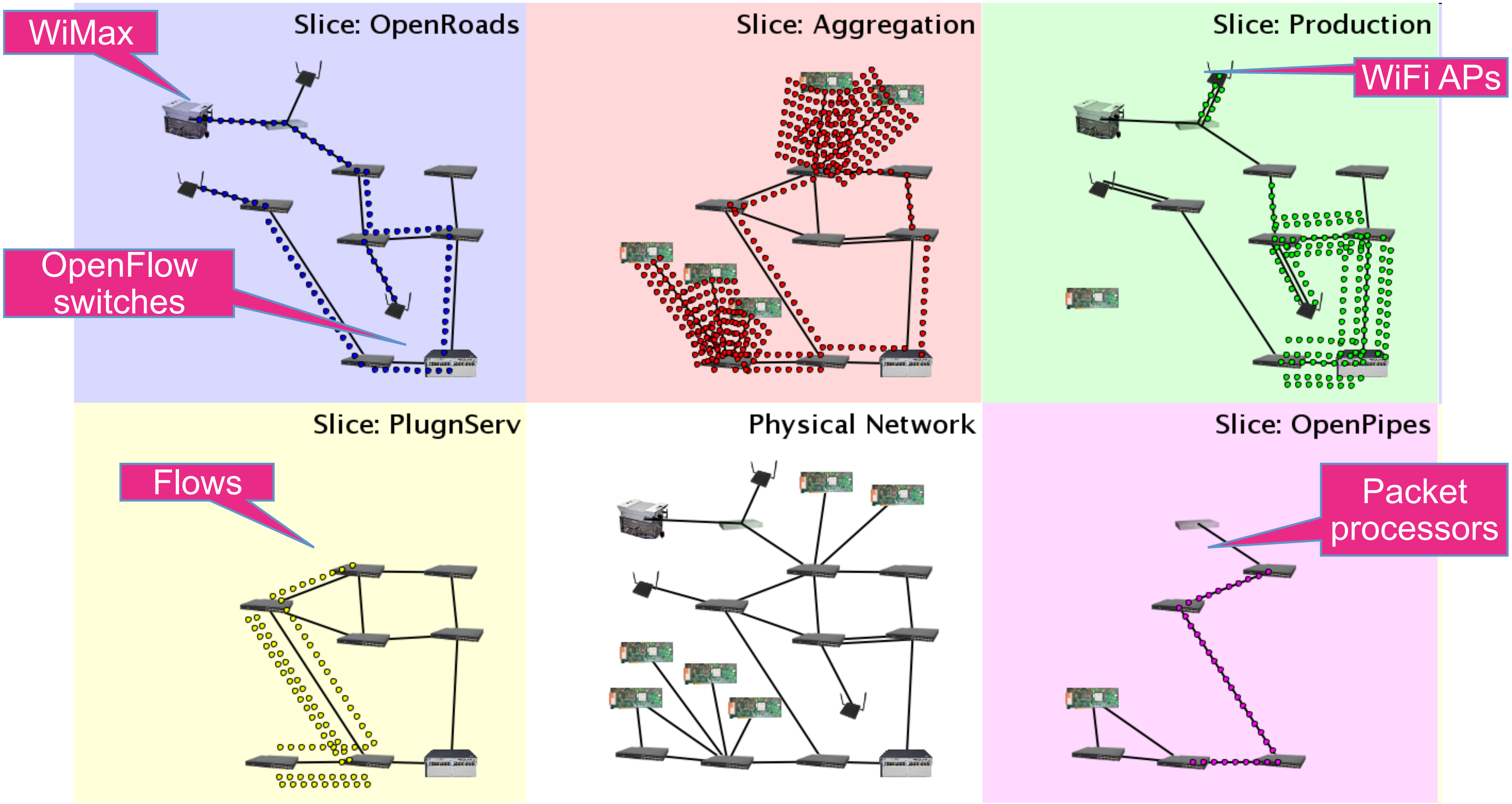
Others

- Removing spanning tree
- Network visualization
- Network debugging
- Packet-circuit convergence
- Home networks
- Flexible access control
- Scale-out routers
- Scale-out data centers

OpenFlow Demonstration Overview

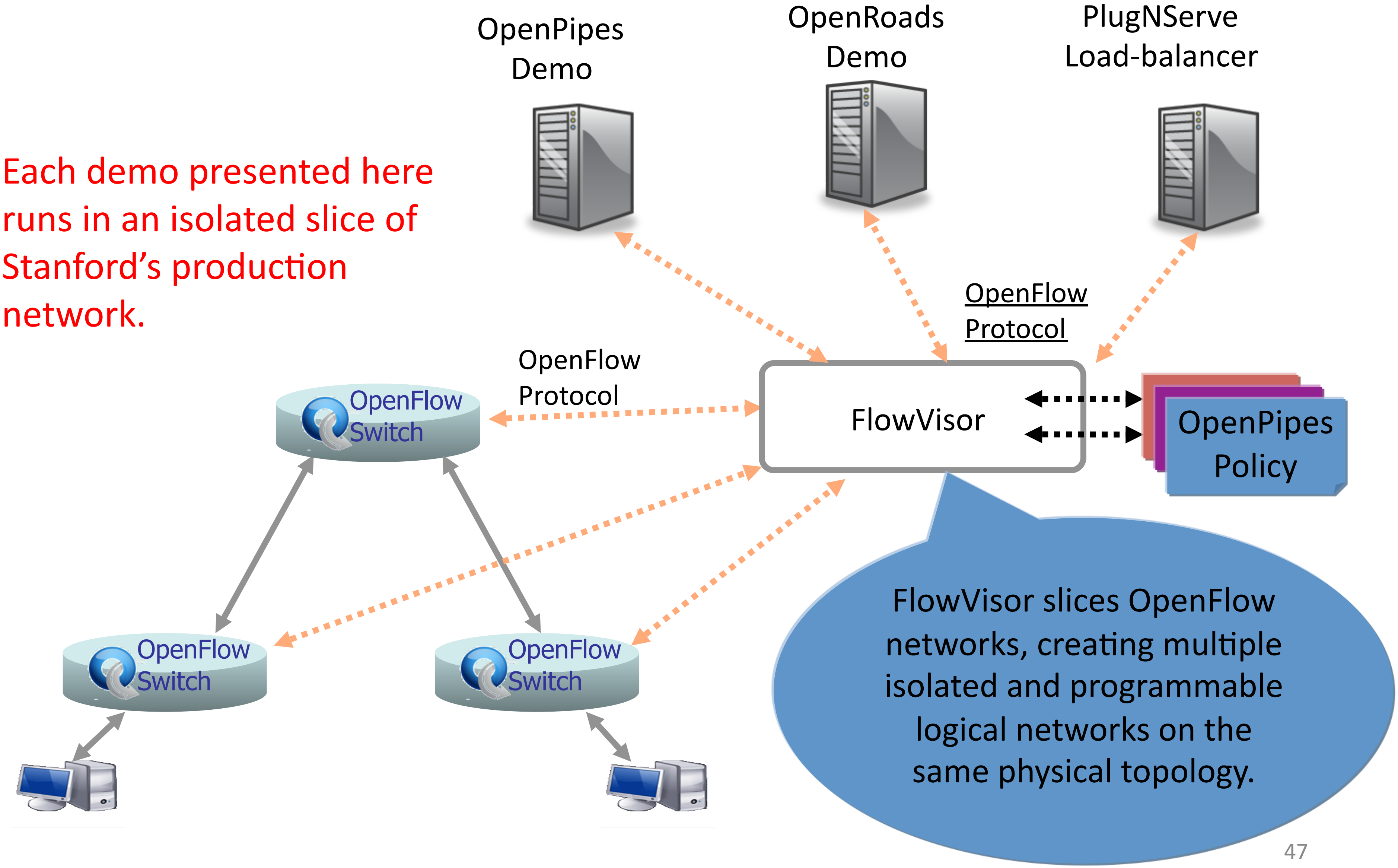
Topic	Demo
Network Virtualization	FlowVisor
Hardware Prototyping	OpenPipes
Load Balancing	PlugNServe
Energy Savings	ElasticTree
Mobility	MobileVMs
Traffic Engineering	Aggregation
Wireless Video	OpenRoads

Demo Infrastructure with Slicing



FlowVisor Creates Virtual Networks

Each demo presented here runs in an isolated slice of Stanford’s production network.



OpenPipes Demo

OpenRoads Demo

PlugNServe Load-balancer

OpenFlow Protocol

OpenFlow Protocol

FlowVisor

OpenPipes Policy

OpenFlow Switch

OpenFlow Switch

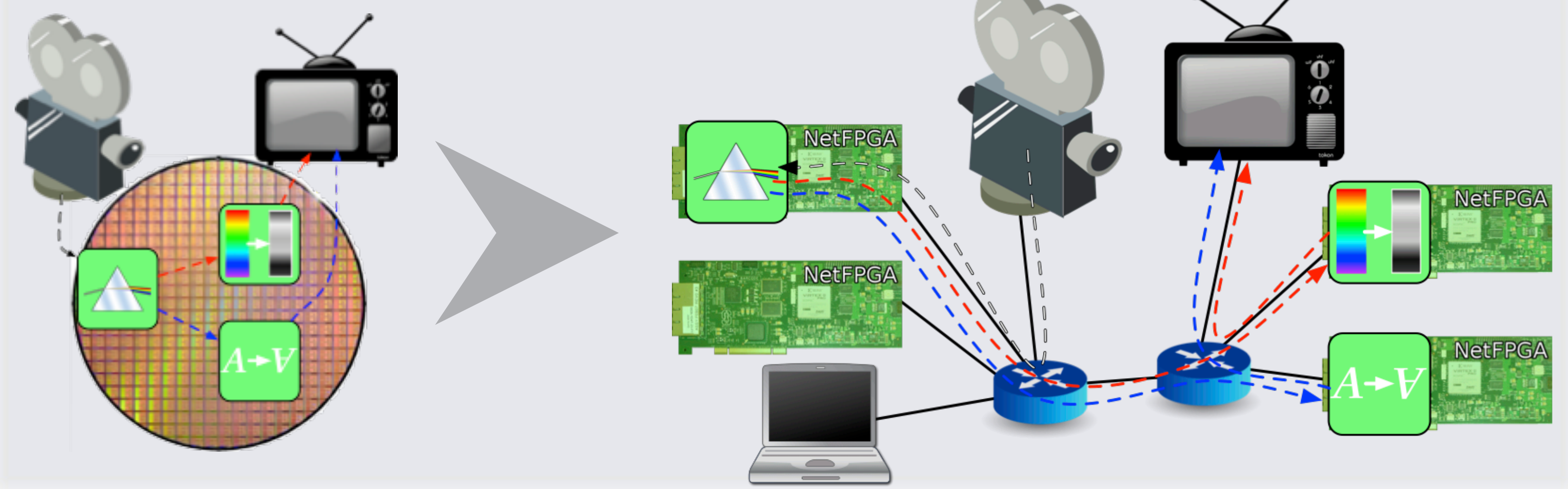
OpenFlow Switch

FlowVisor slices OpenFlow networks, creating multiple isolated and programmable logical networks on the same physical topology.

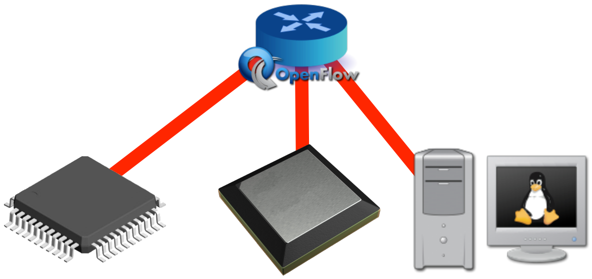
OpenPipes

- Plumbing with OpenFlow to build hardware systems

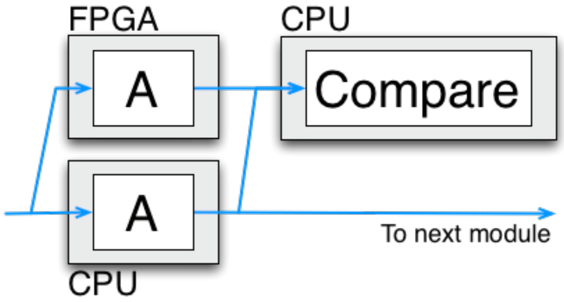
Partition hardware designs



Mix resources

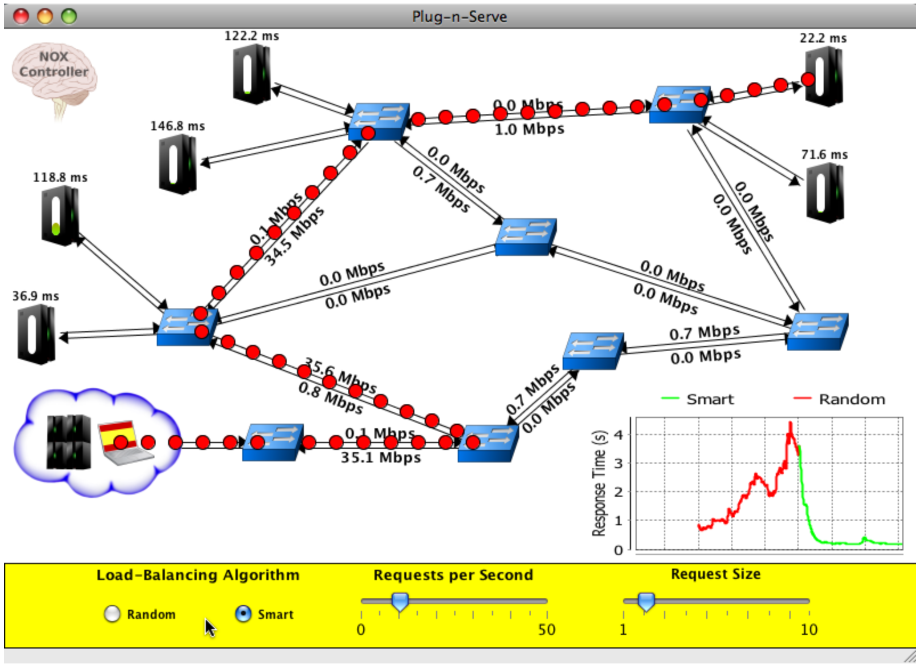


Test



Plug-n-Serve:
Load-Balancing Web Traffic using OpenFlow

Goal: Load-balancing requests in unstructured networks



Load-Balancing Algorithm ○ Random ● Smart

Requests per Second 0 50

Request Size 1 10

What we are showing

- OpenFlow-based distributed load-balancer
  - Smart load-balancing based on network and server load
  - Allows incremental deployment of additional resources

OpenFlow means…

- Complete control over traffic within the network
- Visibility into network conditions
- Ability to use existing commodity hardware

This demo runs on top of the FlowVisor, sharing the same physical network with other experiments and production traffic.

Dynamic Flow Aggregation on an OpenFlow Network

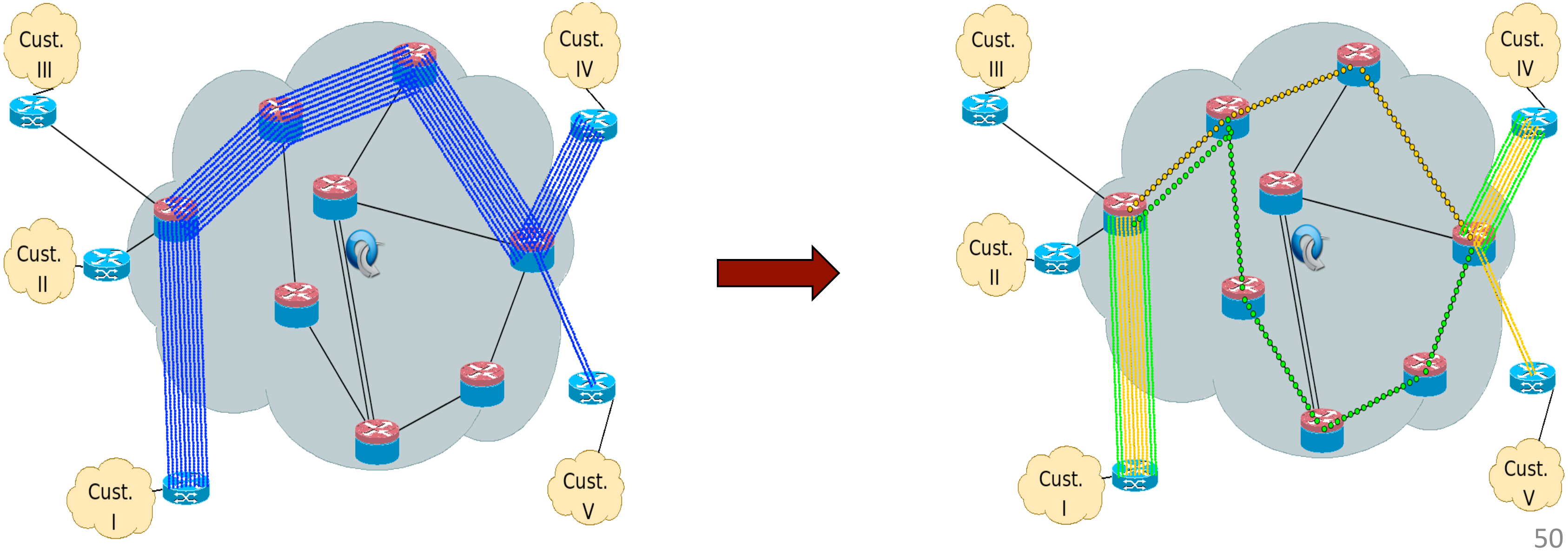
Scope

- Different Networks want different flow granularity (ISP, Backbone,…)
- Switch resources are limited (flow entries, memory)
- Network management is hard
- Current Solutions : MPLS, IP aggregation

How OpenFlow Helps?

- Dynamically define flow granularity by wildcarding arbitrary header fields
- Granularity is on the switch flow entries, no packet rewrite or encapsulation
- Create meaningful bundles and manage them using your own software (reroute, monitor)

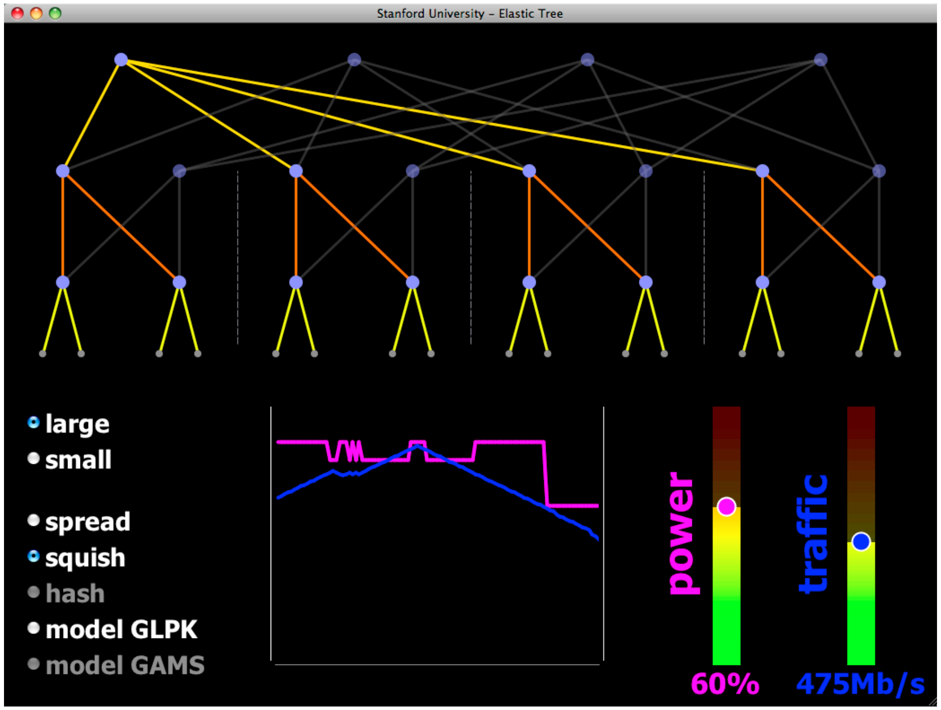
Higher Flexibility, Better Control, Easier Management, Experimentation



ElasticTree:

Reducing Energy in Data Center Networks

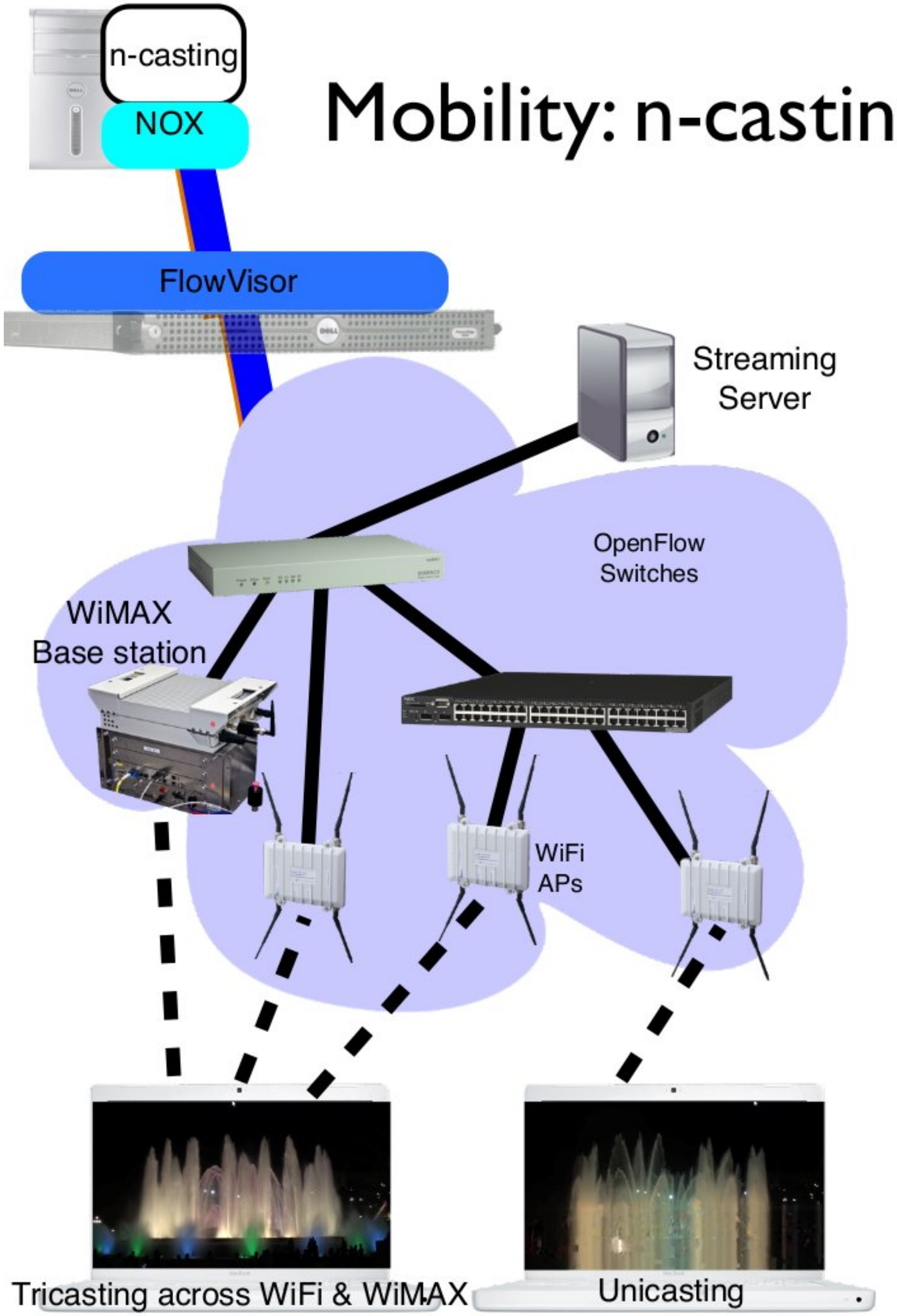
- Shuts off links and switches to reduce data center power
- Choice of optimizers to balance power, fault tolerance, and BW
- OpenFlow provides network routes and port statistics



- The demo:
- Hardware-based 16-node Fat Tree
- Your choice of traffic pattern, bandwidth, optimization strategy
- Graph shows live power and latency variation

demo credits: Brandon Heller, Srini Seetharaman, Yiannis Yiakoumis, David Underhill

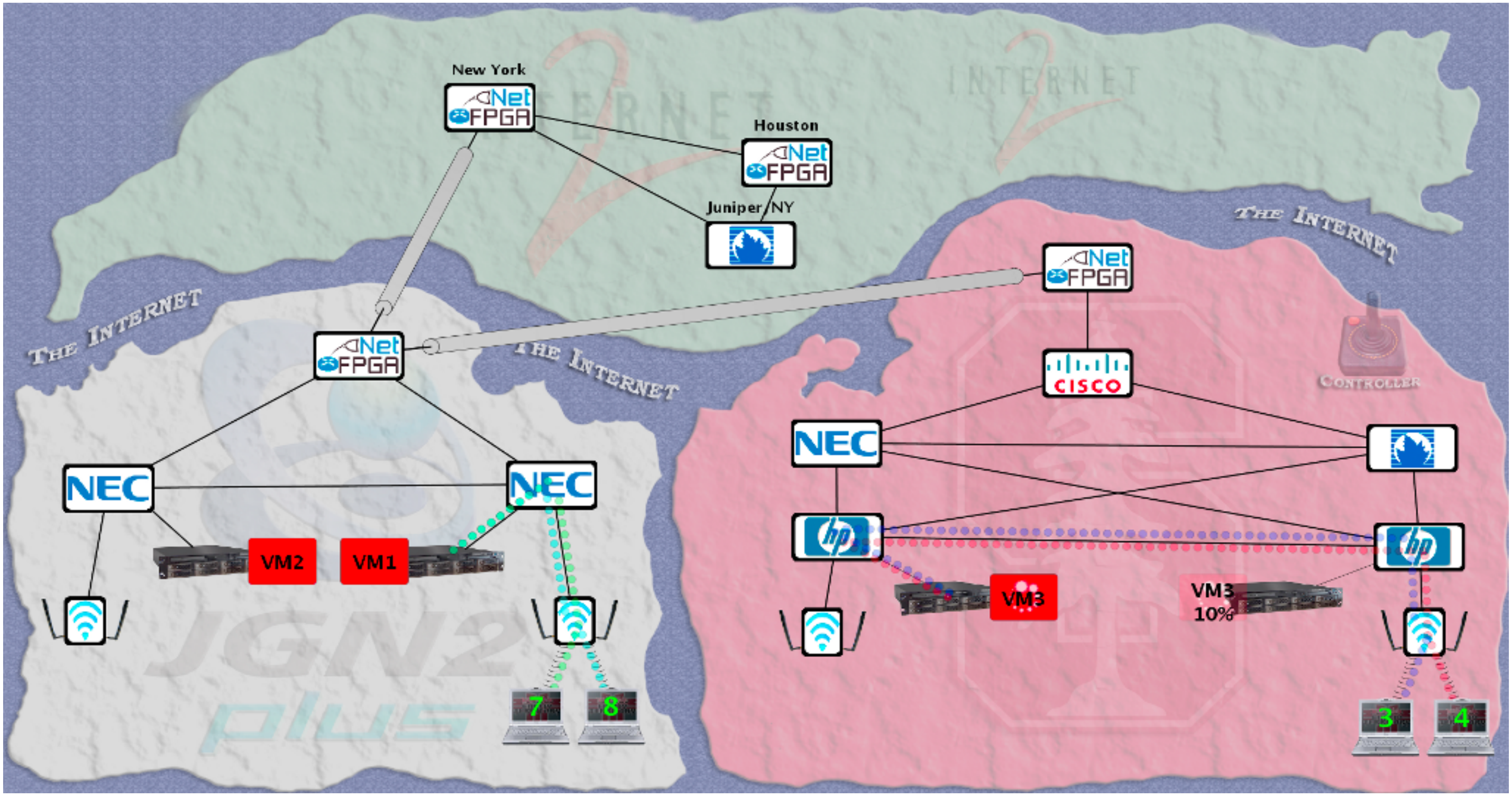
Mobility: n-casting with OpenFlow



- Demonstrate what flexibility of routing enables in mobile networks
- Show how technology agnostic handover can be easily achieved
- Customized network services for applications, devices and technologies
- Simplify control and services
- Unified control for wireline and wireless networking equipments

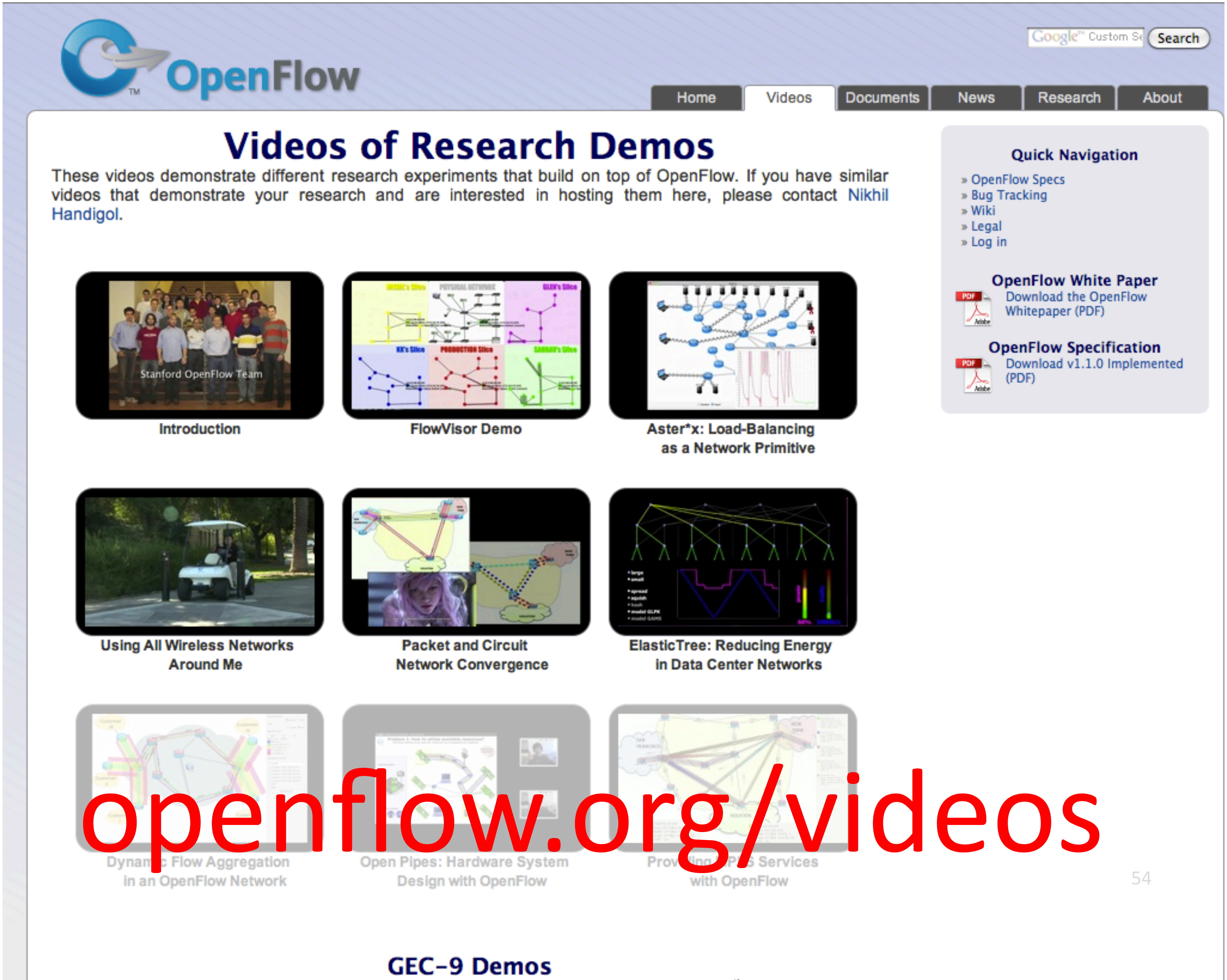
- Demonstration: n-casting
  - Reroute flows between WiFi and WiMAX without additional logic
  - n-casting provided over for video streaming where application handles duplication well
  - coded in 227 lines of C/C++

Intercontinental VM Migration



Moved a VM from Stanford to Japan without changing its IP.

VM hosted a video game server with active network connections.



Videos of Research Demos

These videos demonstrate different research experiments that build on top of OpenFlow. If you have similar videos that demonstrate your research and are interested in hosting them here, please contact Nikhil Handigol.

Introduction

FlowVisor Demo

Aster*x: Load-Balancing as a Network Primitive

Using All Wireless Networks Around Me

Packet and Circuit Network Convergence

ElasticTree: Reducing Energy in Data Center Networks

Dynamic Flow Aggregation in an OpenFlow Network

Open Pipes: Hardware System Design with OpenFlow

Provisioning MPLS Services with OpenFlow

GEC-9 Demos

Quick Navigation

OpenFlow White Paper

Download the OpenFlow Whitepaper (PDF)

OpenFlow Specification

Download v1.1.0 Implemented (PDF)

openflow.org/videos

Video Time

Hands-on Tutorial

Next presentation starts at **1:30**

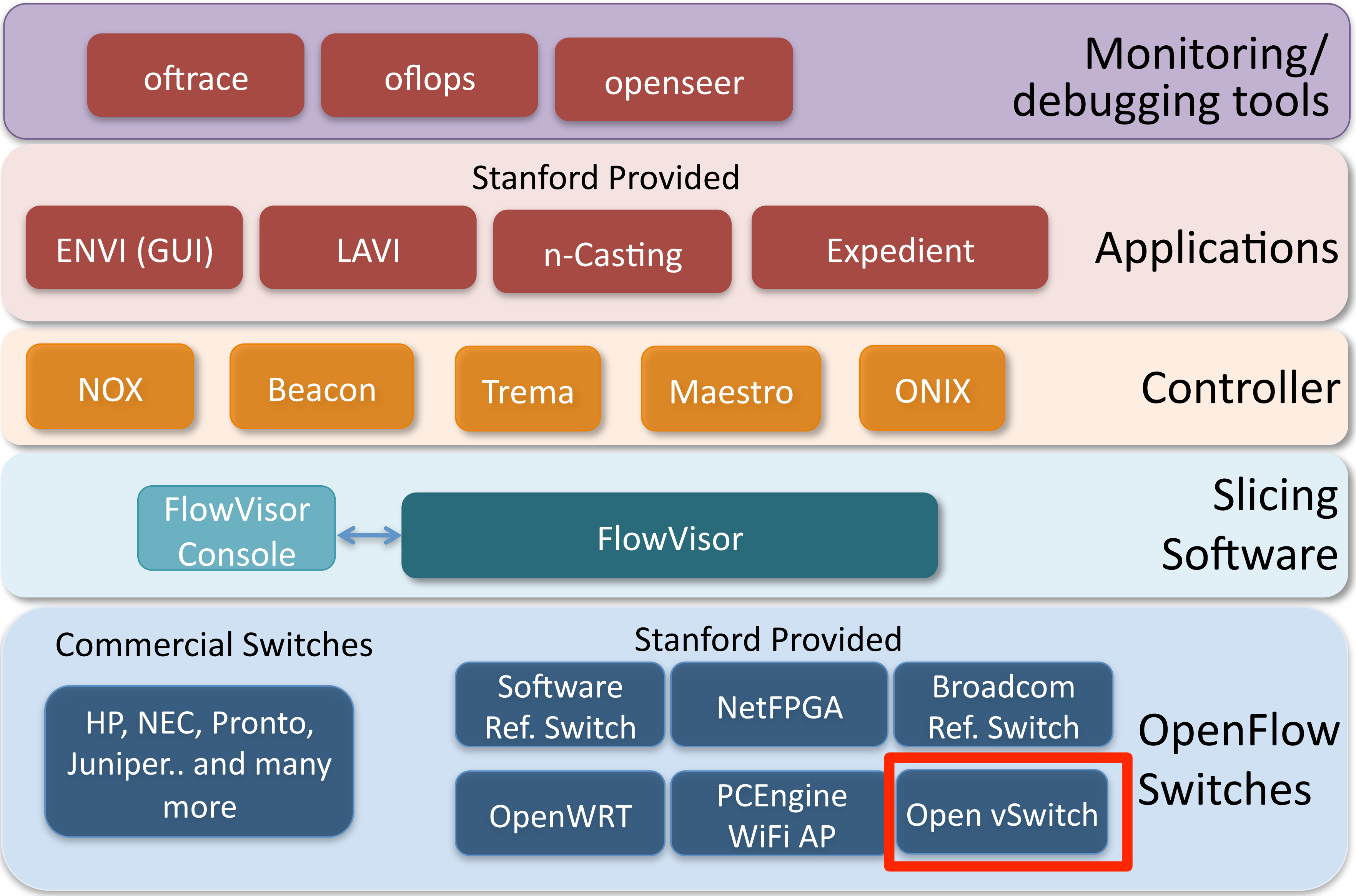
SDN Building Blocks

Instructions still at:

www.openflow.org/wk/index.php/OpenFlow_Tutorial

SDN Building Blocks

OpenFlow building blocks



Monitoring/ debugging tools

oftrace

oflops

openseer

Applications

Stanford Provided

ENVI (GUI)

LAVI

n-Casting

Expedient

Controller

NOX

Beacon

Trema

Maestro

ONIX

Slicing Software

FlowVisor Console

FlowVisor

OpenFlow Switches

Commercial Switches

HP, NEC, Pronto, Juniper.. and many more

Stanford Provided

Software Ref. Switch

NetFPGA

Broadcom Ref. Switch

OpenWRT

PCEngine WiFi AP

Open vSwitch

Current SDN hardware

Juniper MX-series

NEC IP8800

WiMax (NEC)

HP Procurve 5400

Netgear 7324

PC Engines

Pronto 3240/3290

Ciena Coredirector

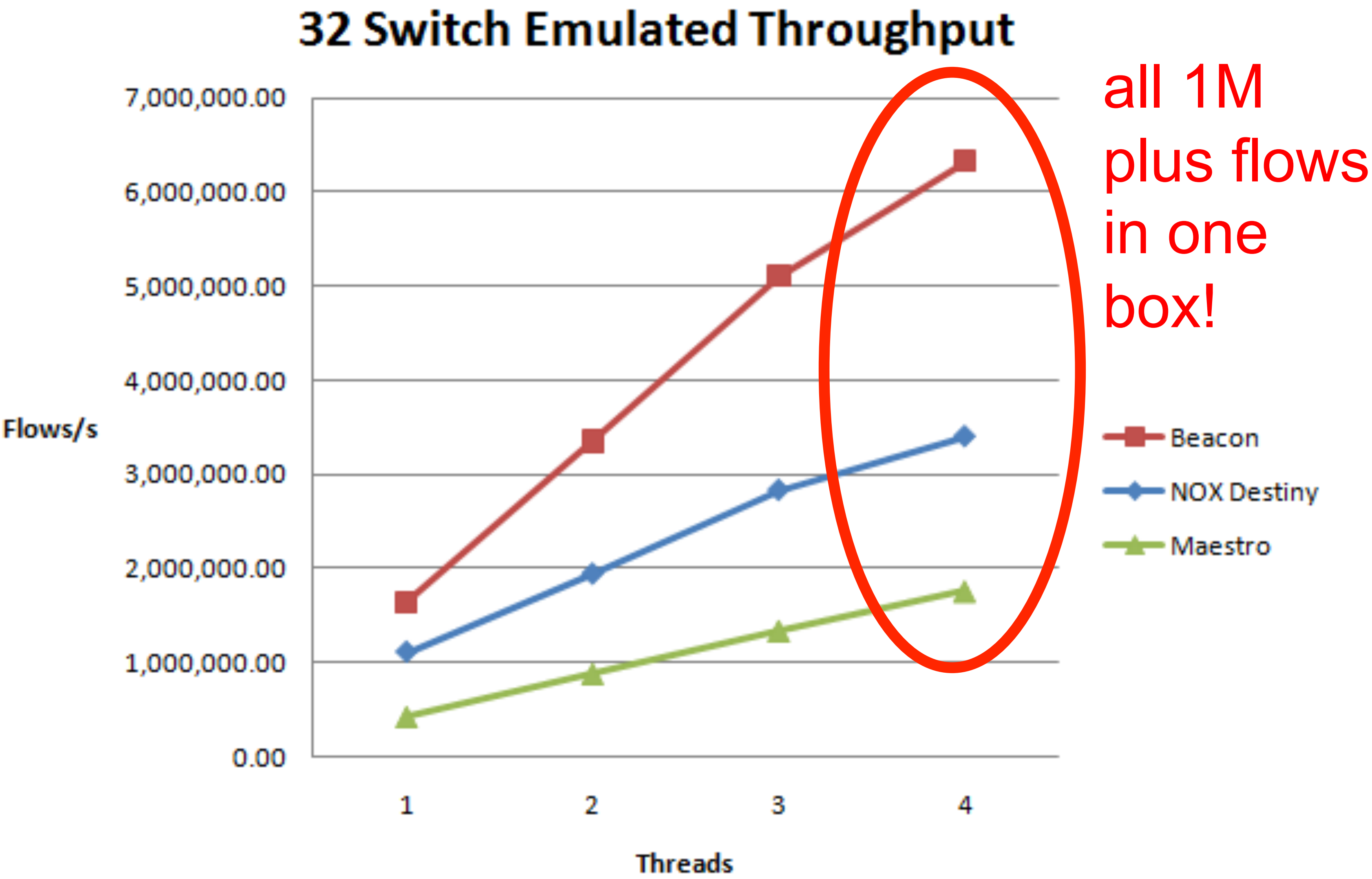
Ask your vendors

Commercial Switch Vendors

Model	Virtualize	Notes	
HP Procurve 5400zl or 6600	1 OF instance per VLAN	-LACP, VLAN and STP processing before OpenFlow -Wildcard rules or non-IP pkts processed in s/w -Header rewriting in s/w -CPU protects mgmt during loop	
NEC IP8800	1 OF instance per VLAN	-OpenFlow takes precedence -Most actions processed in hardware -MAC header rewriting in h/w	
Pronto 3240 or 3290 with Pica8 or Indigo firmware	1 OF instance per switch	-No legacy protocols (like VLAN and STP) -Most actions processed in hardware -MAC header rewriting in h/w	

Open Controllers

Name	Lang	Platform(s)	License	Original Author	Notes
OpenFlow Reference	C	Linux	OpenFlow License	Stanford/ Nicira	not designed for extensibility
NOX	Python, C++	Linux	GPL	Nicira	actively developed
Beacon	Java	Win, Mac, Linux, Android	GPL (core), FOSS Licenses for your code	David Erickson (Stanford)	runtime modular, web UI framework, regression test framework
Maestro	Java	Win, Mac, Linux	LGPL	Zheng Cai (Rice)	
Trema	Ruby, C	Linux	GPL	NEC	includes emulator, regression test framework
RouteFlow	?	Linux	Apache	CPqD (Brazil)	virtual IP routing as a service



all 1M plus flows in one box!

CPU: 1 x Intel Core i7 930 @ 3.33ghz, 9GB RAM, Ubuntu 10.04.1 x64

For more testing details, see:
http://www.openflow.org/wk/index.php/Controller_Performance_Comparisons

Closed-source Controller Vendors

- Ask these vendors for more info
  - BigSwitch
  - Nicira
  - NEC

Growing Community

Vendors and start-ups

BROADCOM.

JUNIPer NETWORKS

ERICSSON

hp invent

NEC

ciena

DELL

NETGEAR® Connect with Innovation™

extreme networks

ARISTA

nicira

FUJITSU

HUAWEI

MARVELL®

big switch networks

More...

Providers and business-unit

BT

Google™

verizon

INTERNET2®

NLR light the future

Level(3)℠ COMMUNICATIONS

NTT docomo

Microsoft®

facebook.

amazon.com®

More...

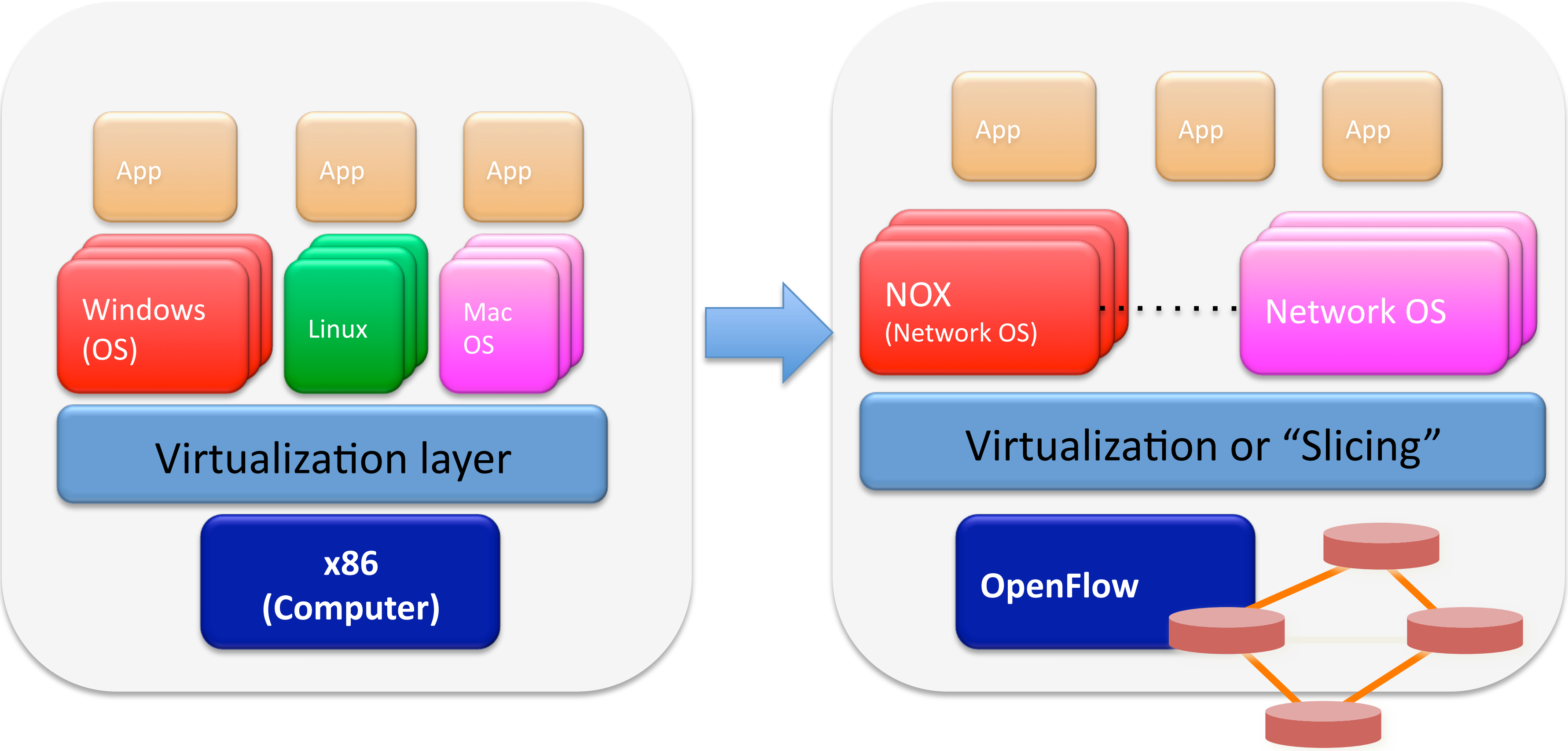
Note: Level of interest varies

Related Research

- DIFANE
  - Rule partitioning for controller-less flow insertion
- UCSD Fat Tree Series: Scalable Commodity Data Center, PortLand, Hedera
  - Scale-out data centers that use OpenFlow
- Tesseract
  - Centralized WAN in the 4D Architecture
- ONIX
  - Fault-tolerant controller platform from Nicira, Google, NEC
- DevoFlow
  - Practical scalability limits to OpenFlow and modifications to get around them

Virtualizing OpenFlow

Trend



App App App

Windows (OS) Linux Mac OS

Virtualization layer

x86 (Computer)

Computer Industry

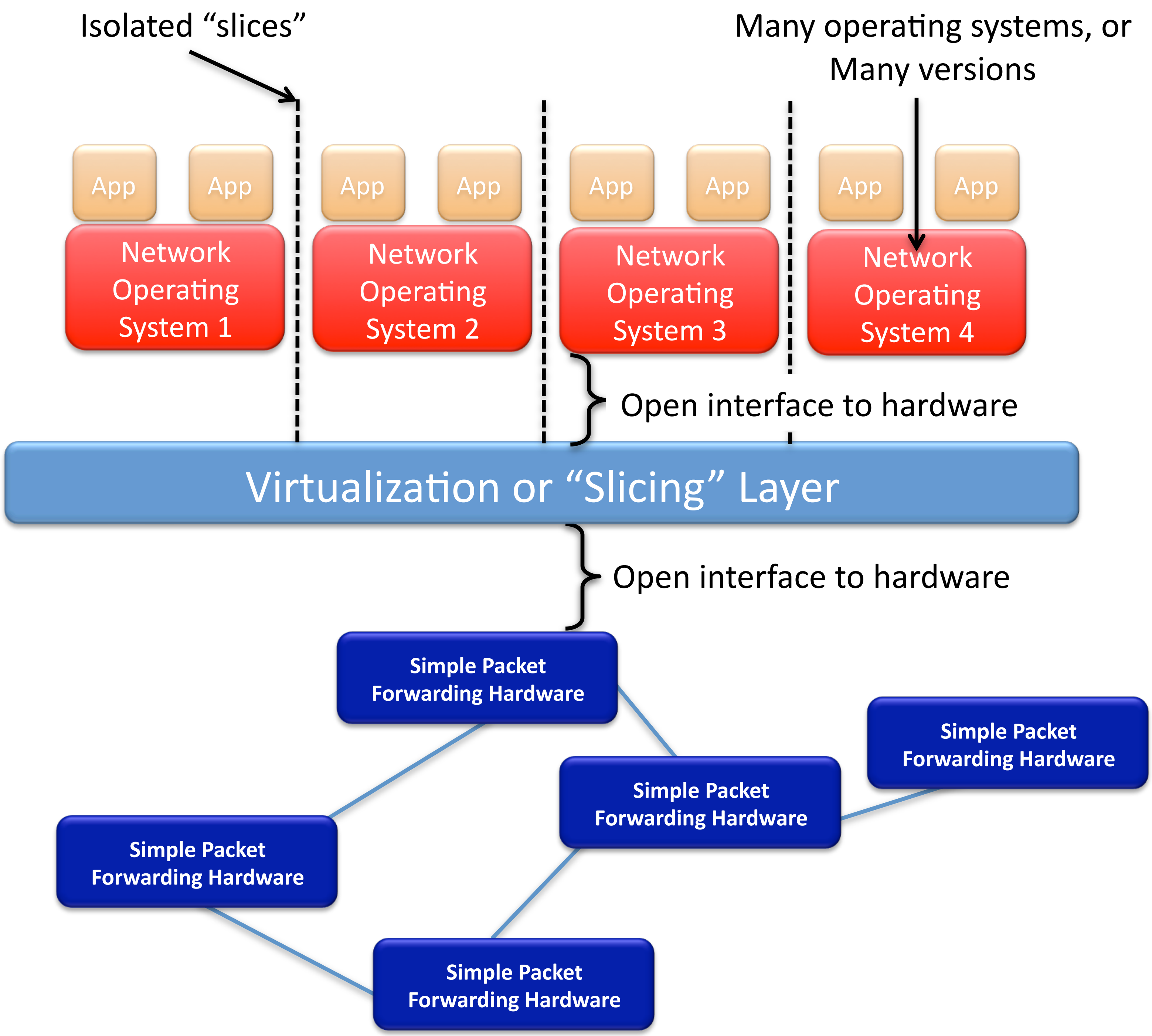
App App App

NOX (Network OS) Network OS

Virtualization or “Slicing”

OpenFlow

Network Industry



Isolated “slices”

Many operating systems, or
Many versions

App App App App App App App App

Network Operating System 1

Network Operating System 2

Network Operating System 3

Network Operating System 4

Open interface to hardware

Virtualization or “Slicing” Layer

Open interface to hardware

Simple Packet Forwarding Hardware

Simple Packet Forwarding Hardware

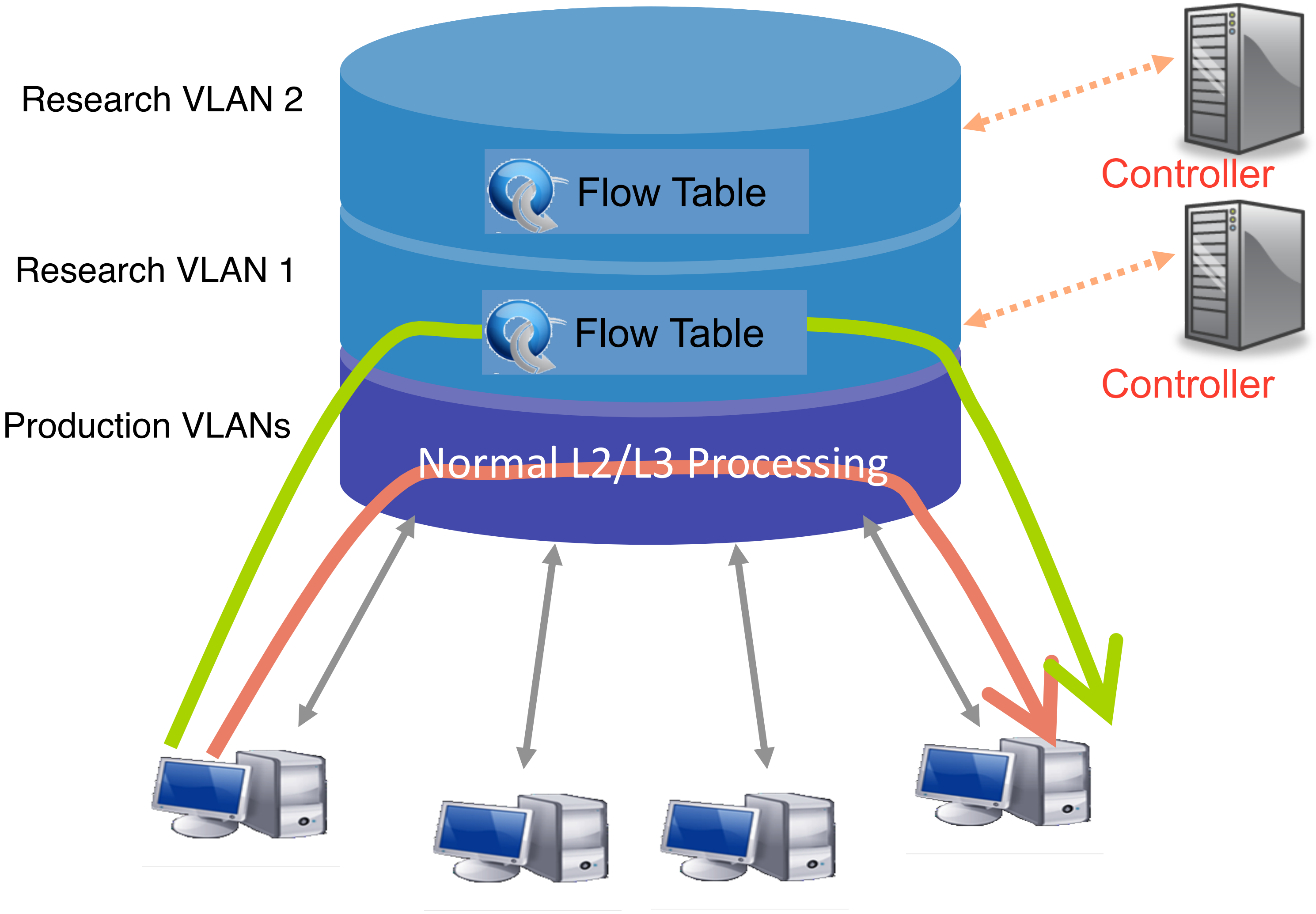
Simple Packet Forwarding Hardware

Simple Packet Forwarding Hardware

Simple Packet Forwarding Hardware

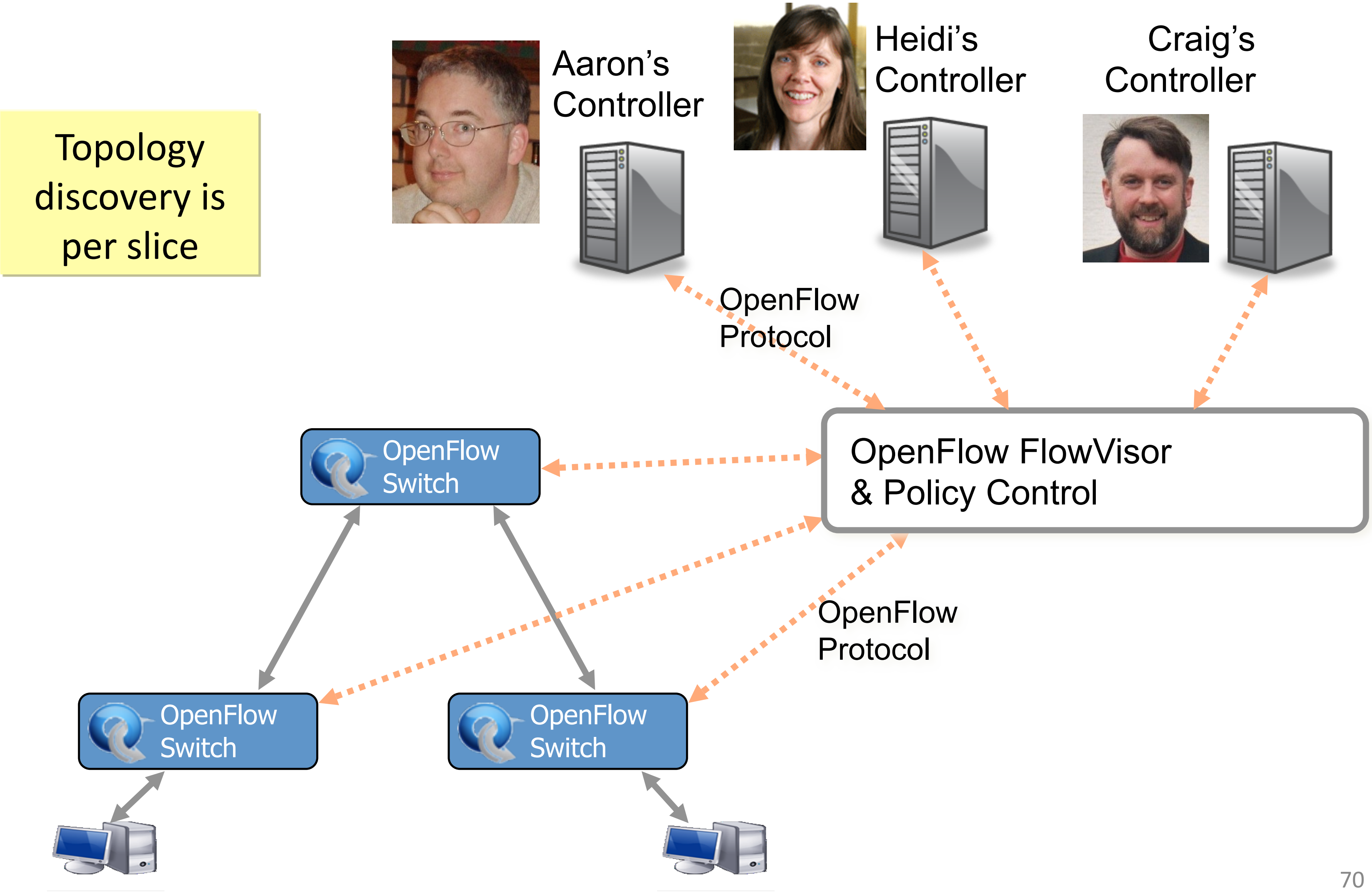
Switch Based Virtualization

Exists for NEC, HP switches but not flexible enough



FlowVisor-based Virtualization

Topology discovery is per slice



Aaron's Controller

Heidi's Controller

Craig's Controller

OpenFlow Protocol

OpenFlow FlowVisor & Policy Control

OpenFlow Switch

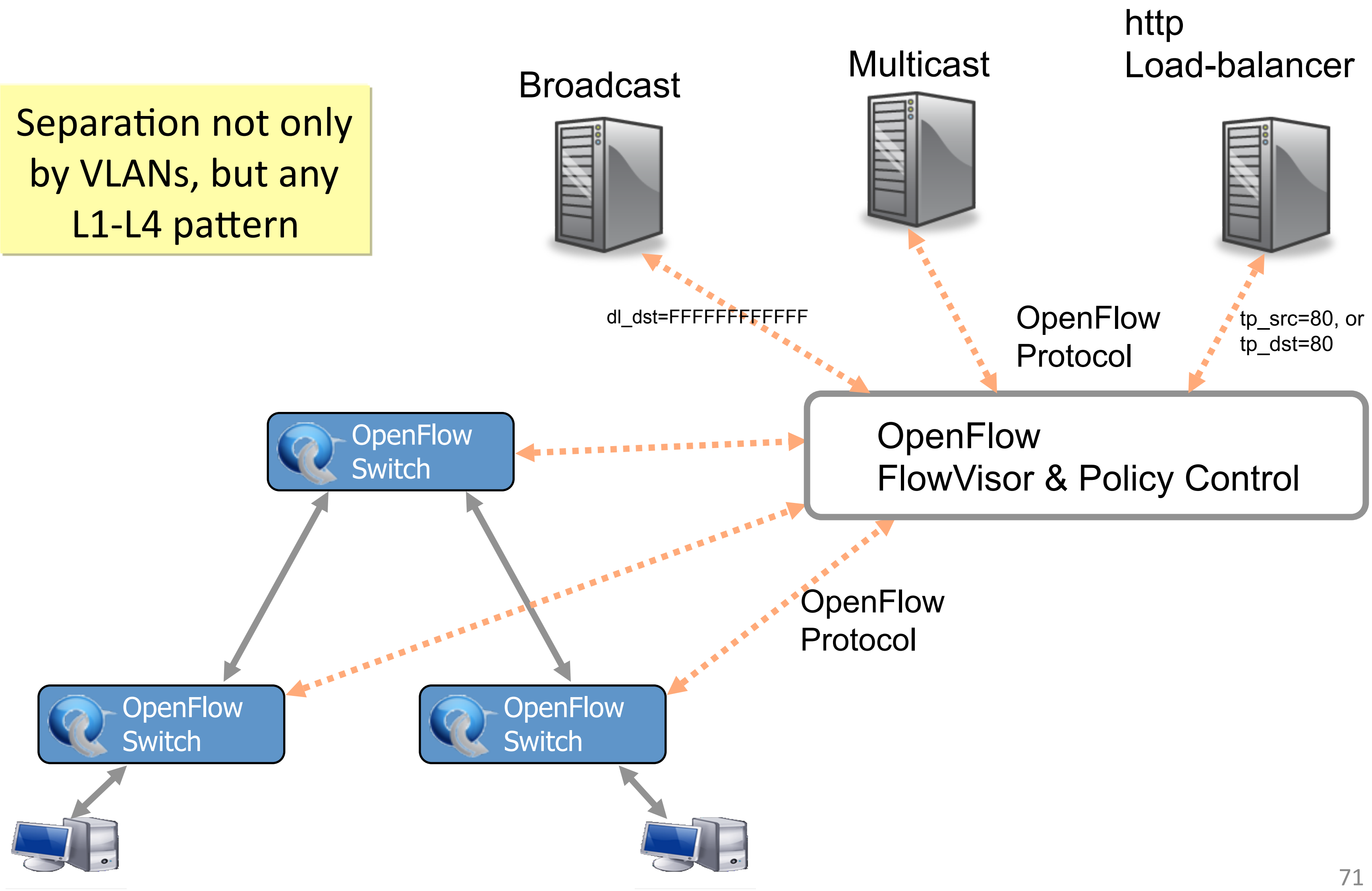
OpenFlow Switch

OpenFlow Switch

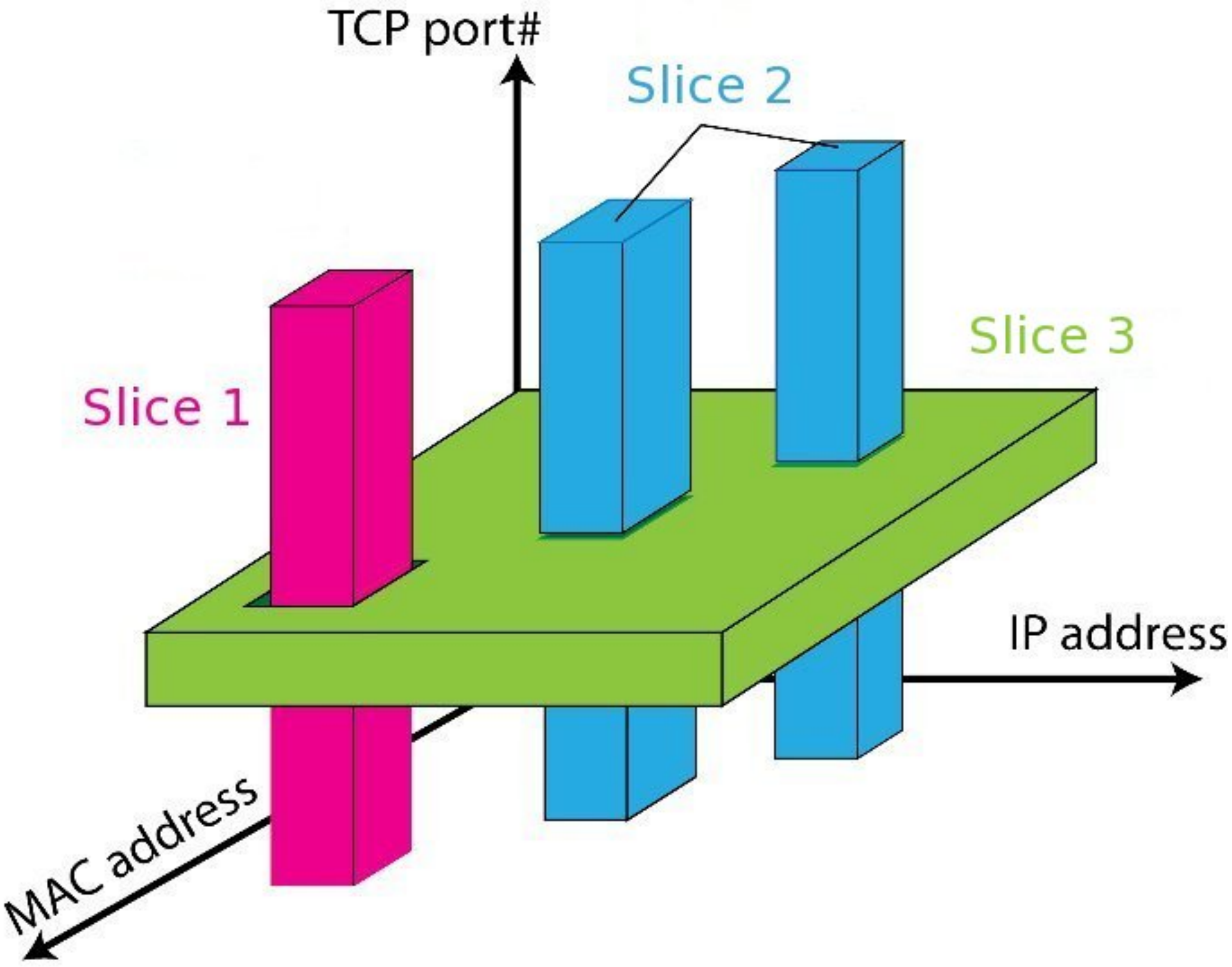
OpenFlow Protocol

FlowVisor-based Virtualization

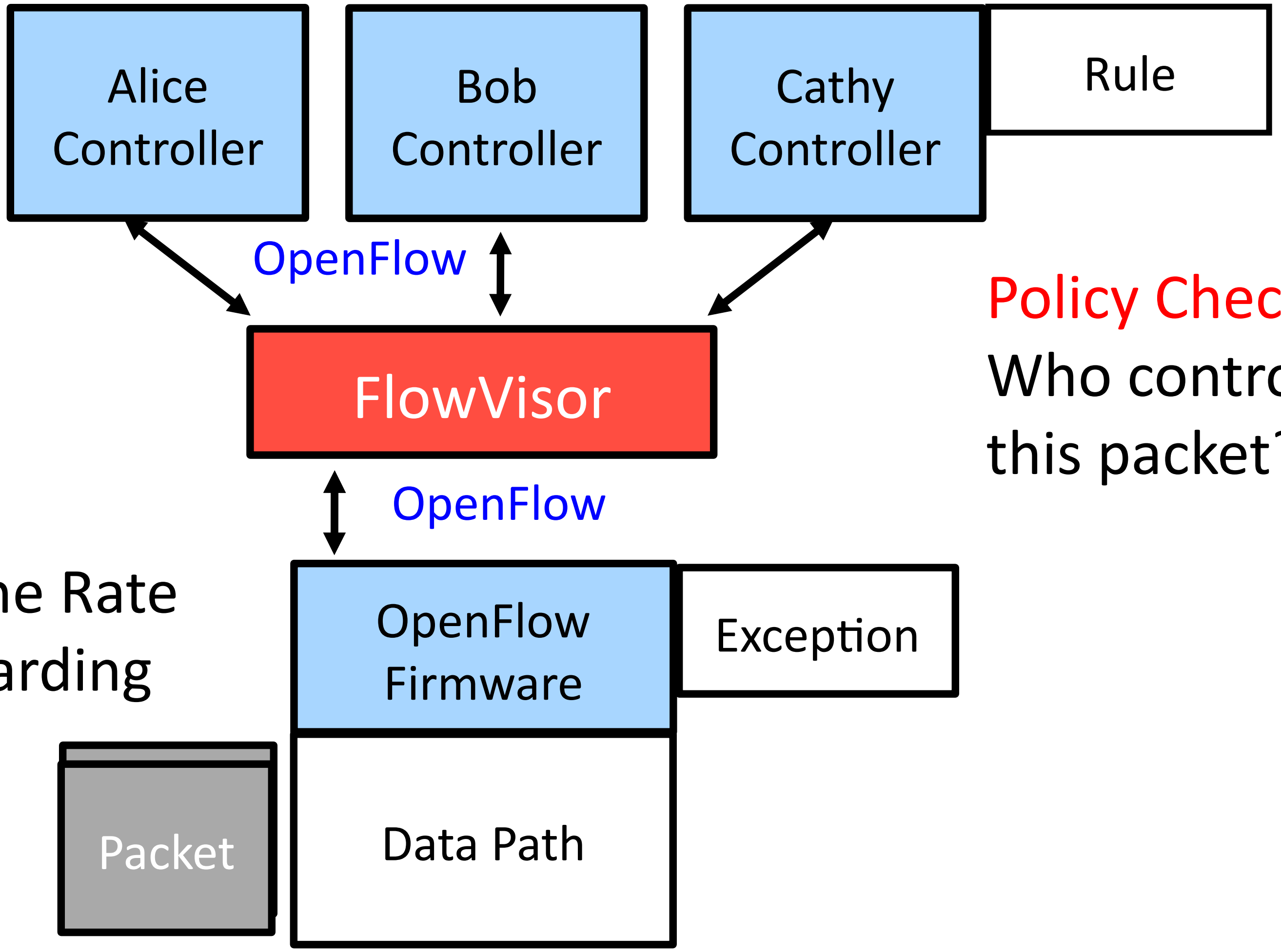
Separation not only by VLANs, but any L1-L4 pattern



FlowSpace: Maps Packets to Slices



FlowVisor Message Handling



Policy Check:
Is this rule allowed?

Policy Check:
Who controls this packet?

Use Case: New CDN - Turbo Coral ++

Basic Idea: Build a CDN where you control the entire network

- All traffic to or from Coral IP space controlled by Experimenter
- All other traffic controlled by default routing
- Topology is entire network
- End hosts are automatically added (no opt-in)

	Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport
	*	*	*	*	*	84.65.*	*	*	*	*
	*	*	*	*	*	*	84.65.*	*	*	*
	*	*	*	*	*	*	*	*	*	*

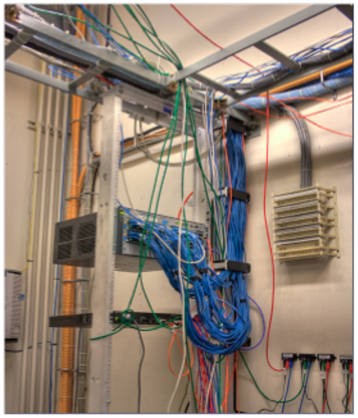
Use Case: Aaron’s IP

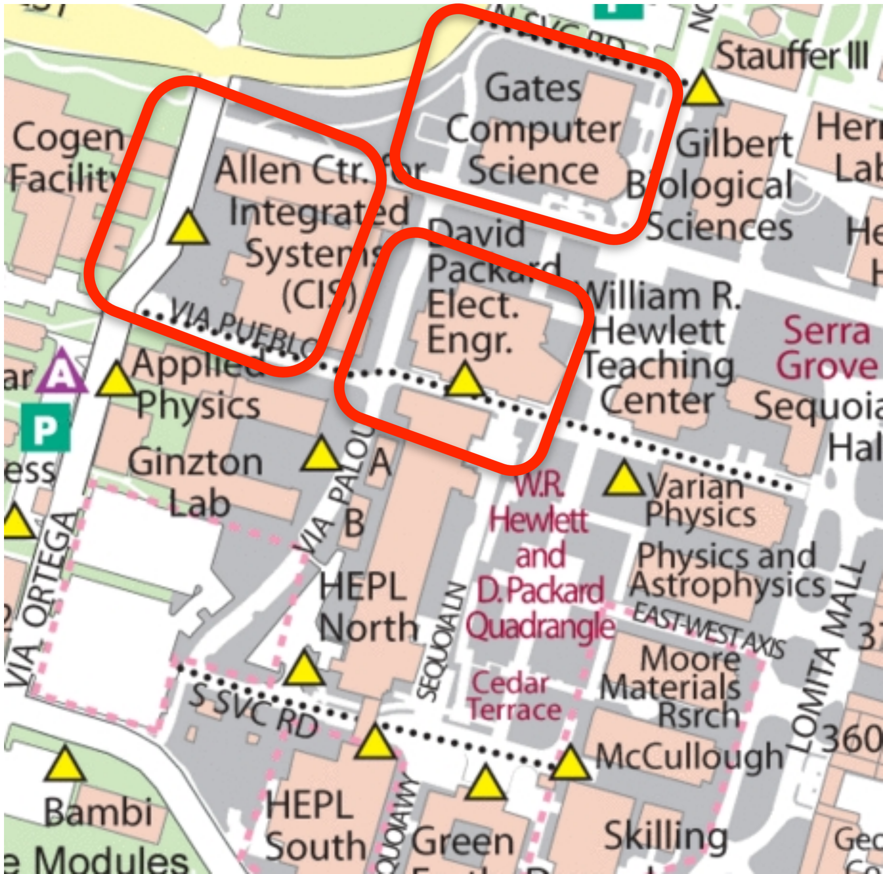
- A new layer 3 protocol
- Replaces IP
- Defined by a new Ether Type

	Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport
	*	*	*	AaIP	*	*	*	*	*	*
	*	*	*	!AaIP	*	*	*	*	*	*

OpenFlow Deployment at Stanford

Switches (23)
APs (50)
WiMax (1)





Live Stanford
Deployment Statistics

http://yuba.stanford.edu/ofhallway/wide-ofv1.html

How did we get there?

Staged Deployment of OpenFlow

1. Add experimental VLAN
2. Enable OpenFlow for Exptl VLAN
3. Configure Controller for new network

Verify correctness and performance

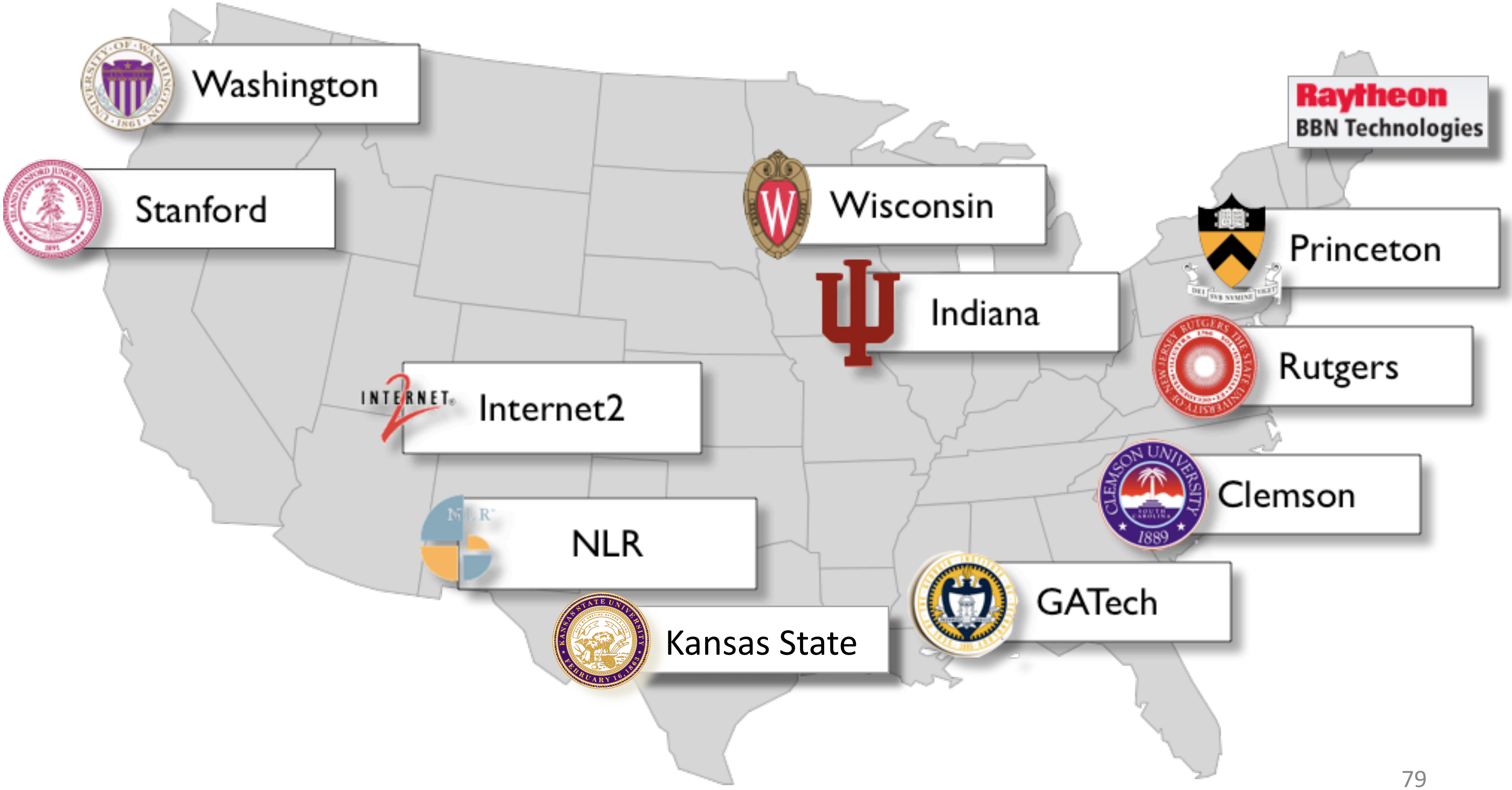
4. Add new Production subnetwork
5. Gradually add/move users to new subnet

Verify reachability

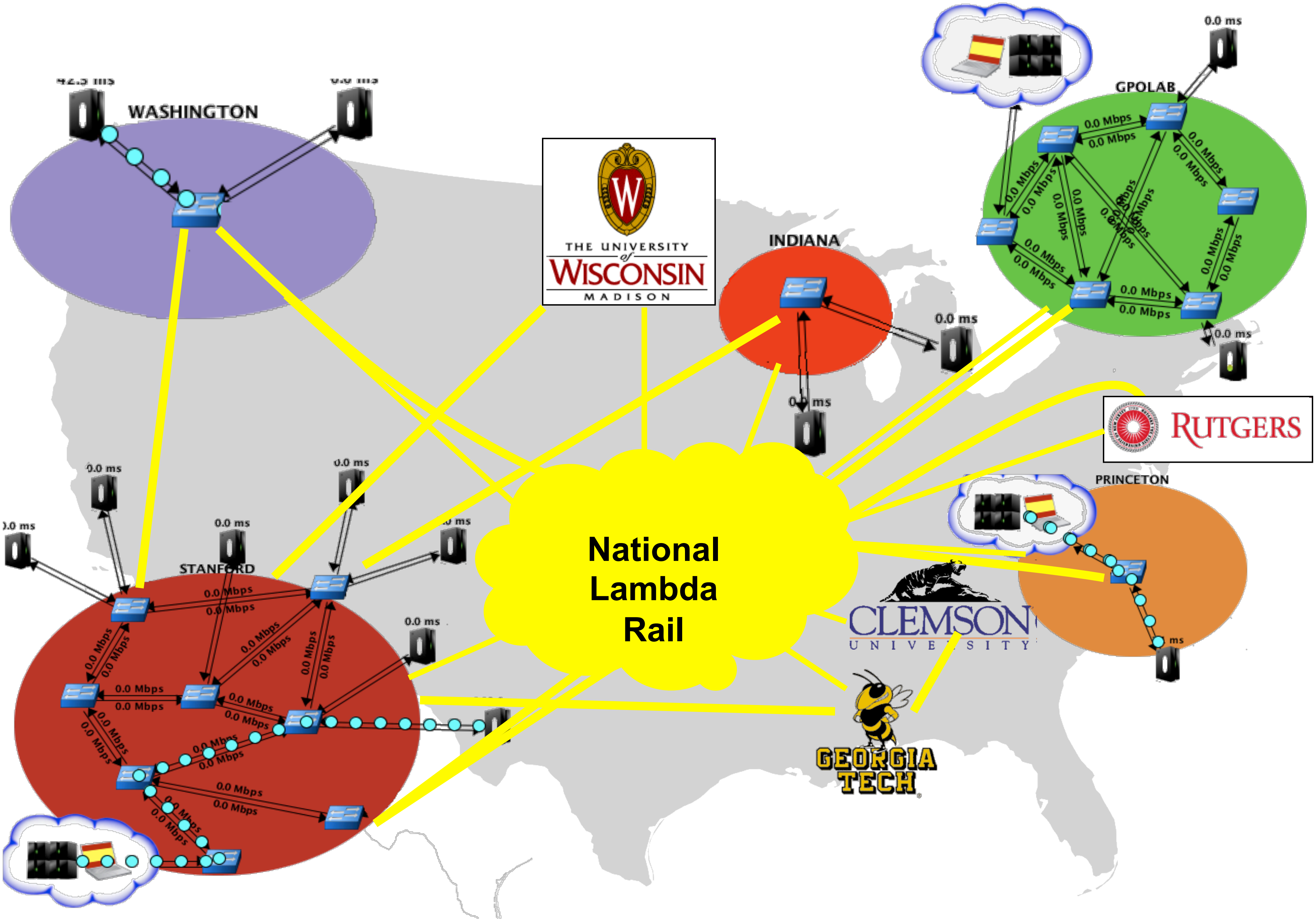
6. Enable OpenFlow for new subnet
7. Slice the network

GENI OpenFlow deployment (2010)

10 institutions and 2 National Research Backbones

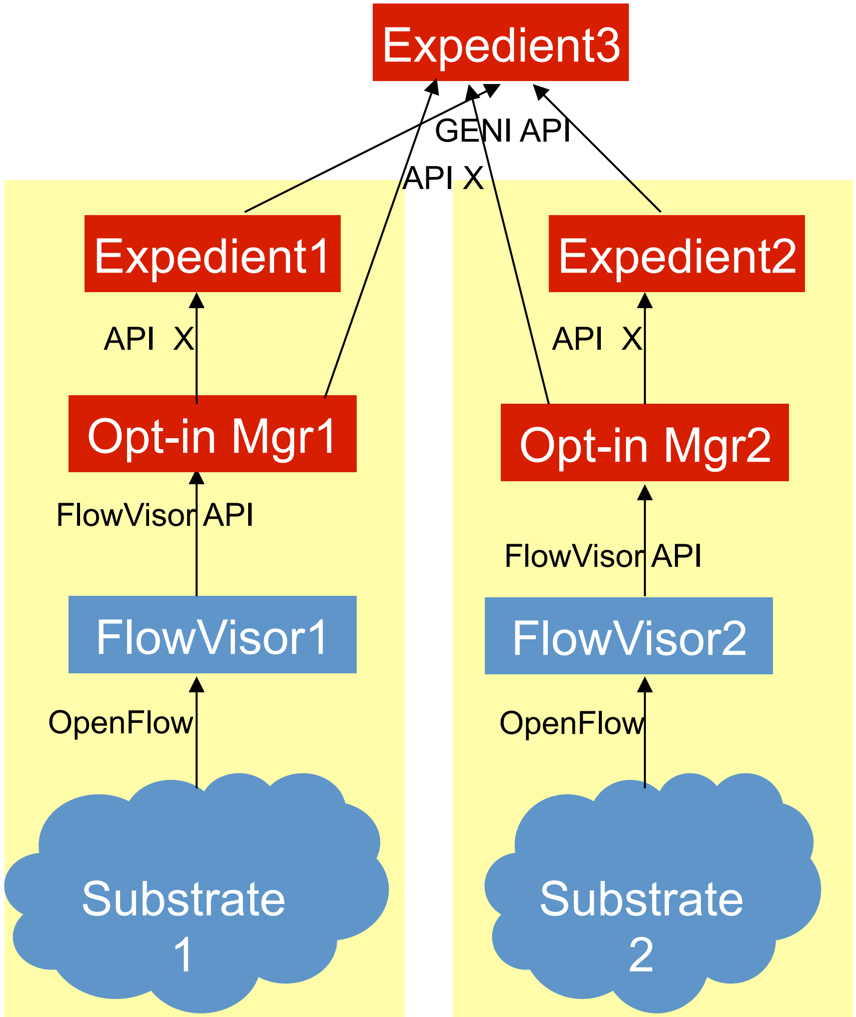


GENI Network Evolution



GENI Integration

- FlowVisor
  - Slicing control
- Expedient
  - Experimenter's portal for slice management
- Opt-in Manager
  - Network admins' portal to approve/ deny expt requests for traffic



Mistakes we made

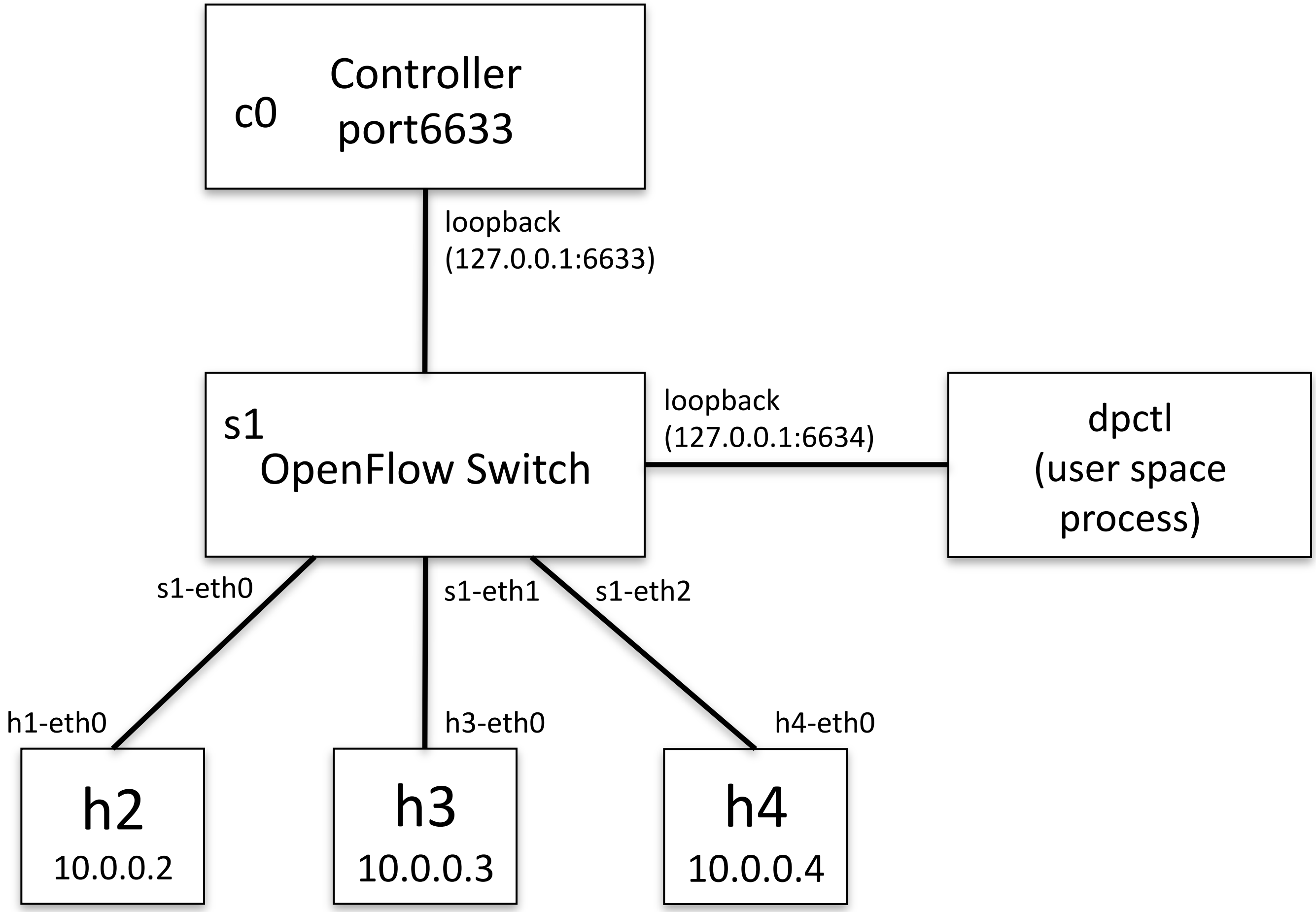
- OpenFlow over Q-in-Q
  - OpenFlow routing is unaware and sends traffic with same MAC address in both direction, causing perpetual learning and CPU inflation
- Moving uplinks for 1 switch, while being pointed to the same controller (causing two islands)
  - Causes controller learning to oscillate between the 2 uplinks
- Bad interaction with legacy protocols
  - LLDP and STP are treated differently with different switches
- Loop in OpenFlow network being exposed to non-OF side
  - Miscommunication between the aggregate operator and the experimenter during testing phase
- Loop across backbones
  - Same campus connected over NLR and Internet2

Next steps for GENI

- Remove duct-tape
  - Fix any issues that we learned about during demos
- Be better prepared
  - Higher stability and better isolation
  - Test bandwidth slicing
- Grow topology
  - Add more switches and hosts
- Wean off
  - Each campus takes charge of control framework

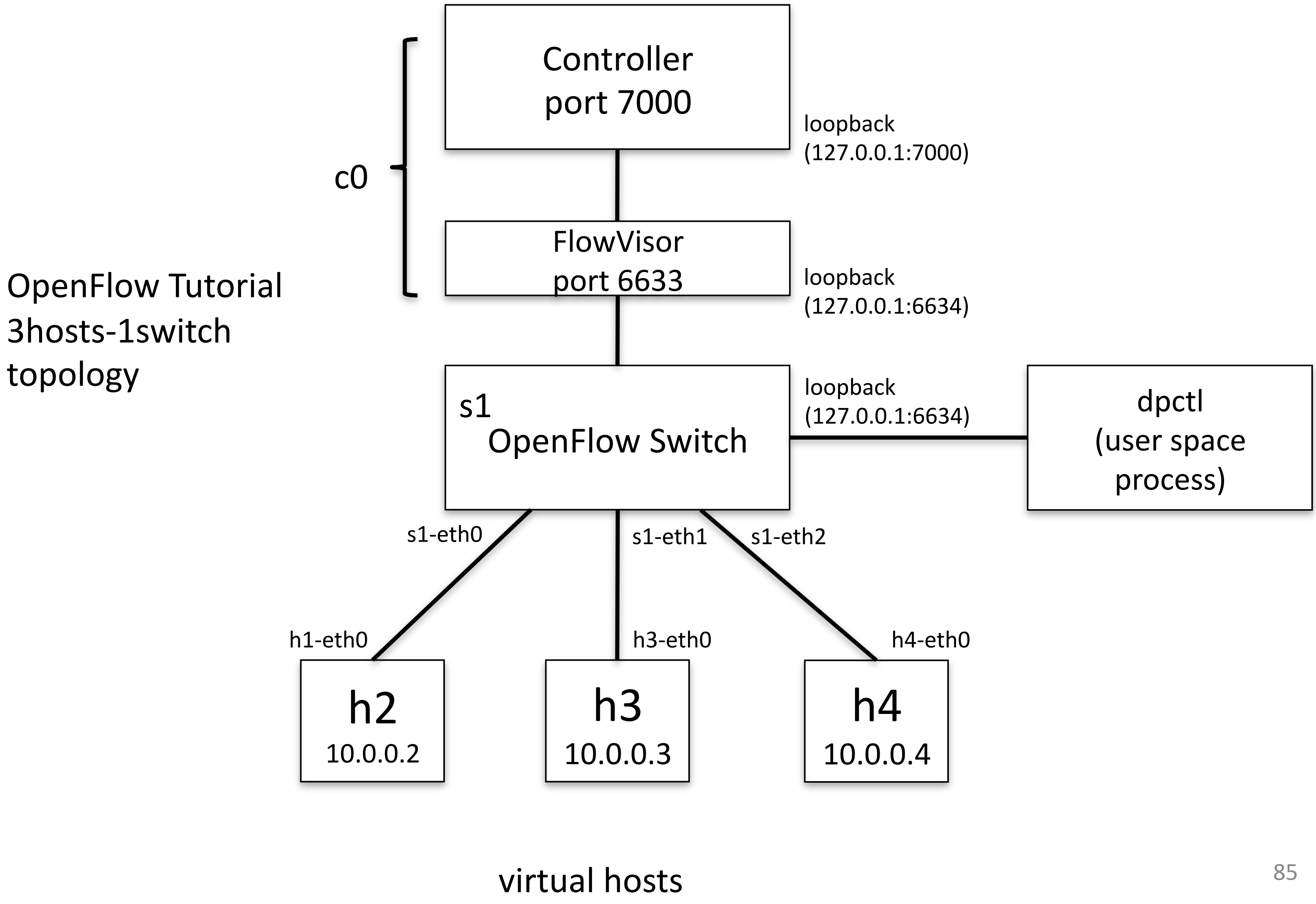
Tutorial Setup so far

OpenFlow Tutorial
3hosts-1switch
topology



virtual hosts

Virtualized Network Setup



FlowVisor Usage

- man ./doc/flowvisor.8
- <flowvisor dir>/script/fvctl.sh
- listDevices
  - list of all OpenFlow switches' datapath ID
- getLinks
  - list of all links (port # and datapath ID of both end switches)
- createSlice
  - creating slice (specifying controller's URL and slice name)
- listSlices
- addFlowSpace
  - add flow space to slices
- listFlowSpace
  - show current flow space

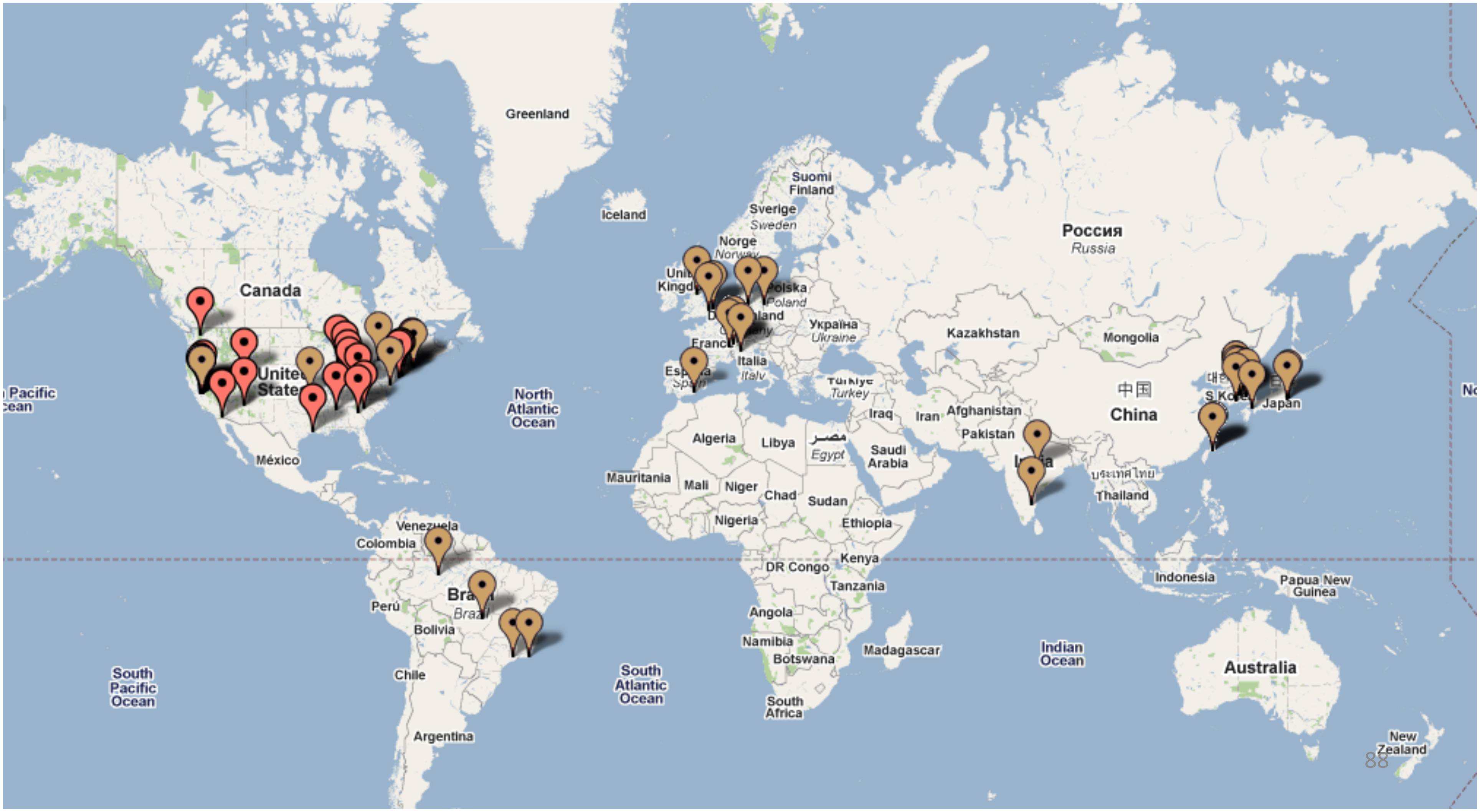
Highlights of Deployments

- Stanford deployment
  - McKeown group for 1.5 years: production and experiments
  - To scale later this year to entire building (~500 users)
- Nation-wide trials and deployments
  - 7 other universities and BBN deploying now
  - GEC9 in Nov, 2010 showcased nation-wide OF
  - Internet 2 and NLR starting to serve as the GENI Backbone
- Global trials
  - Over 60 organizations experimenting

2011 likely to be a big year for OpenFlow

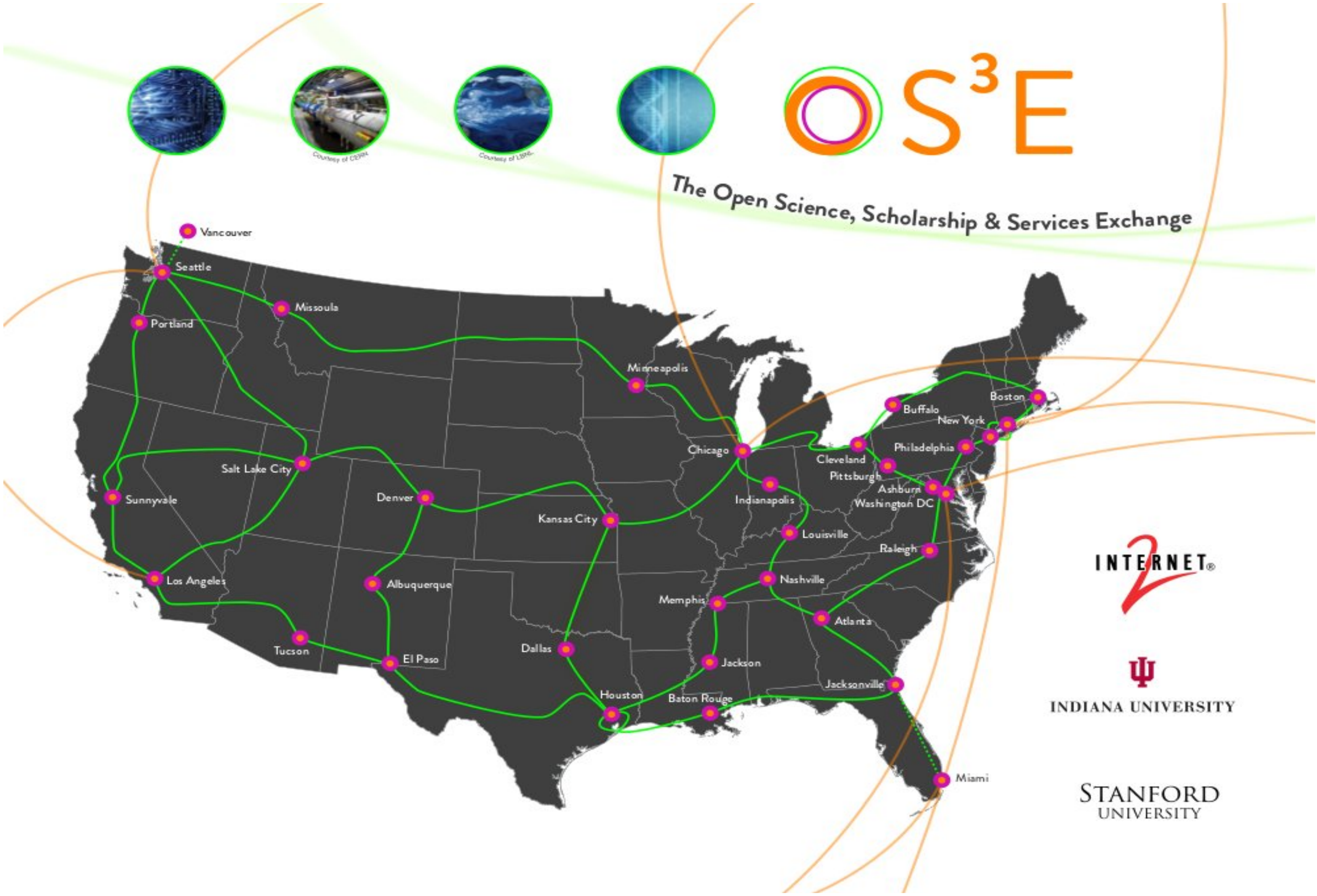
Current Trials

- 68 trials/deployments spanning 13 countries



Internet2 OpenFlow deployment initiative.

35+ 100G POPs, nationwide.



Hands-on Tutorial

Next presentation starts at **2:45**

SDN Deployment Forum

Instructions still at:

www.openflow.org/wk/index.php/OpenFlow_Tutorial

SDN Deployment Forum

Our speakers

- Johan van Reijendam (Stanford)
- David Erickson (Stanford)
- Subhasree Mandal (Google)

Questions

- What was your hardest/funniest error to debug?
- What's your one strongest gripe about OpenFlow?
- Do you have a simple example of something that your network does better now?

Closing

Closing Thoughts

- OpenFlow is a protocol
  - How to use it is up to you
- Software-Defined Networking is an architecture
  - Of which OpenFlow is just a piece
- OpenFlow is available, used, and improving

- These are the early stages for OF, ONS, and SDN

Get involved!

- Ask and answer questions on mailing lists:
  - openflow-discuss
  - openflow-spec
- Join and participate Open Networking Forum
- Share and update wiki content
- Submit bug-reports and/or patches to OF reference implementation and Open vSwitch
- Release open-source applications
- Write a controller!

Thanks!

SDN Team at Stanford